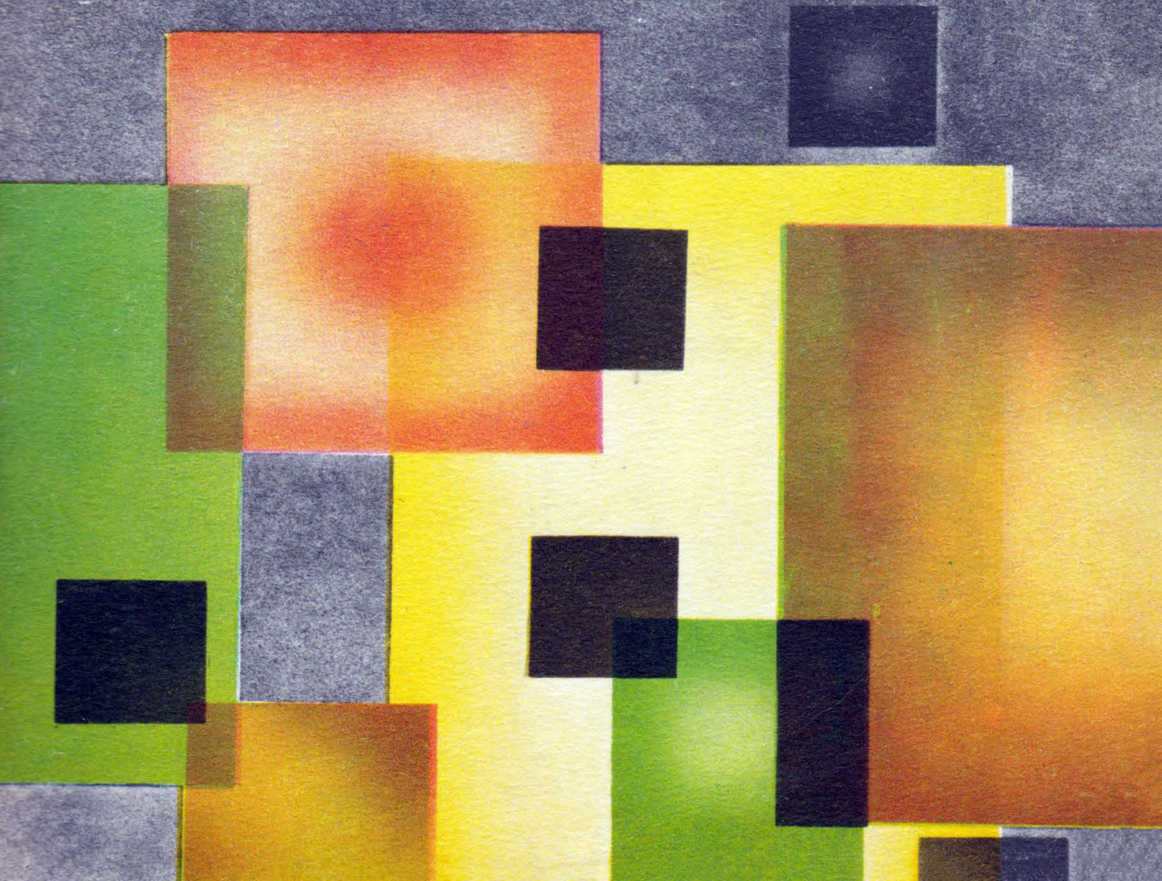
CRISTIAN LUPU ȘTEFAN STĂNCESCU

MICROPROCESOARE

CIRCUITE • PROIECTARE



MICROPROCESOARE

CIRCUITE • PROIECTARE

CRISTIAN LUPU ȘTEFAN STĂNCESCU

MICROPROCESOARE

CIRCUITE • PROIECTARE

EDITURA MILITARĂ, BUCUREȘTI, 1986

Coperta: V. ILIE
Ilustrație: GH. DUNĂREANU

CUVÎNT ÎNAINTE

Apreciind rolul deosebit de important al electronicii în dezvoltarea economiei românești în următoarea etapă, în Directivele Congresului al XIII-lea al Partidului Comunist Român se apreciază că industria electronică va pătrunde masiv în întreaga activitate economico-socială, cu o orientare preponderentă spre dezvoltarea producției de componente electronice, mijloace de automatizare, echipamente de electronică industrială și profesională.

Realizările obținute în domeniile automatizării, electronizării, robotizării materializează preocuparea constantă a partidului și statului nostru pentru introducerea tehnicilor de vîrf în cele mai diverse sectoare de activitate, ce se caracterizează în esență prin orientarea strategiei economice în direcția creșterii rolului factorilor intensivi în dezvoltarea economiei românești. „Avînd în vedere tendințele generale ale dezvoltării mondiale, cît și forța industriei noastre socialiste — *sublinia tovarășul Nicolae Ceaușescu, secretar general al Partidului Comunist Român* —, se impune să acordăm o atenție deosebită noii revoluții tehnico-științifice, astfel încît industria românească, întreaga economie să se ridice la nivelul celor mai noi și avansate cuceriri ale științei și tehnicii contemporane"[1].

Microprocesoarele — exponente de vîrf ale electronicii actuale — constituie elemente principale utilizate în acțiunile de automatizare, electronizare, robotizare. Mai mult, produsele realizate cu microprocesoare, în general produsele de microelectronică, evoluează rapid, demonstrînd o continuă creștere a performanțelor. Aplicarea lor se face, de asemenea, într-un ritm accelerat, în domenii din cele mai diverse, suscitînd interesul unei categorii din ce în ce mai largi de utilizatori.

Punînd în evidență elemente de noutate în domeniu, prezentul volum își propune să adîncească problematica legată de utilizarea microprocesoarelor într-o multitudine de specialități tehnice. Sînt descrise noi circuite LSI, precum și aspecte specifice de proiectare a microcalculatoarelor cum sînt cele referitoare la conectarea unităților de discuri flexibile. Se insistă asupra proiectării modulare a microcalculatoarelor, manieră ce poate conduce la creșterea eficienței în aplicarea tehnicii microprocesoarelor.

Am apreciat utilă prezentarea în primul rînd a familiei de circuite a microprocesorului Z80, considerat ca avînd performanțe deosebite în comparație cu celelalte microprocesoare de aceeași categorie. În continuare s-au descris circuitele

---

[1] Nicolae Ceaușescu, *România pe drumul construirii societății socialiste multilateral dezvoltate*, vol. 28, Editura Politică, București, 1985, p. 34.

specifice pentru cuplarea unităţilor de discuri flexibile la microcalculatoare, cunoscut fiind rolul important al acestor echipamente periferice, care, asociate cu sistemele de operare, oferă posibilitatea implementării unor aplicaţii complexe. Pentru a veni în sprijinul utilizatorilor, am prezentat un exemplu de microcalculator modular realizat pe baza circuitelor descrise. În aceeaşi idee au fost incluse şi anexele care cuprind programe, precum şi schemele unui microcalculator personal realizat cu Z80.

În elaborarea lucrării am căutat să-i conferim acesteia un pronunţat caracter practic pentru a o face cît mai utilă specialiştilor direct implicaţi în realizarea sau folosirea dispozitivelor de comandă cu microprocesoare, echipamentelor pentru conducerea şi dirijarea armamentului şi tehnicii de luptă, frecvent întîlnite în aviaţie, marină, radiolocaţie, transmisiuni, alte domenii.

Se pune astfel la dispoziţia inginerilor electronişti, automatişti, ciberneticienilor, atît militari cît şi civili, un instrument eficient în procesul de proiectare, realizare şi exploatare a variatelor dispozitive construite cu microprocesoare.

Autorii aduc mulţumiri tuturor celor care i-au sprijinit în definitivarea lucrării, inginerului Eugen Mărăcineanu pentru elaborarea schemelor calculatorului prezentat în Anexa E. Cu acest prilej exprimăm încă o dată mulţumirile noastre Editurii Militare pentru înţelegerea, solicitudinea manifestate pe tot parcursul pregătirii lucrării.

AUTORII

1

FAMILIA DE CIRCUITE Z80

Familia de circuite Z80 conține unitatea centrală de tipul UC-Z80 și diverse dispozitive de intrare/ieșire, I/E, de uz general. Funcțiile clasice ale unui sistem cu microprocesor (I/E paralelă, I/E serie, numărare/temporizare și acces direct la memorie) se pot implementa ușor cu circuite din această familie, conectînd la UC-Z80 circuitele: PIO-Z80, pentru comanda I/E paralele, SIO-Z80 sau DART-Z80, pentru comanda I/E serie, CTC-Z80, pentru realizarea funcțiilor de numărare/temporizare, și DMA-Z80 pentru comanda accesului direct la memorie.

Vom descrie în continuare componentele mai des utilizate ale familiei Z80: UC-Z80, PIO-Z80, CTC-Z80 și SIO-Z80.

1.1. UNITATEA CENTRALĂ PE 8 BIȚI UC-Z80

Unitățile centrale de tipul Z80 (Z80, Z80A, Z80B, Z80L) sînt considerate microprocesoare într-o singură capsulă, făcînd parte din generația a treia, cu performanțe superioare în comparație cu celelalte microprocesoare de 8 biți din aceeași categorie.

În figura 1.1 este prezentată schema bloc a unui microprocesor Z80. Procesorul e organizat în jurul unei magistrale interne avînd ca elemente de bază o unitate aritmetică-logică pe 8 biți, UAL, registre, circuitele de comandă a magistralelor de date și adrese, registrul de instrucțiuni împreună cu circuitele pentru decodificarea și comanda unității centrale. Prelucrarea fiecărei instrucțiuni începe cu extragerea ei din memorie și încărcarea în registrul de instrucțiuni. După decodificare, circuitele de comandă generează toate semnalele necesare pentru citirea/scrierea datelor din/în registre, comandă UAL și asigură toate semnalele de comandă ex-

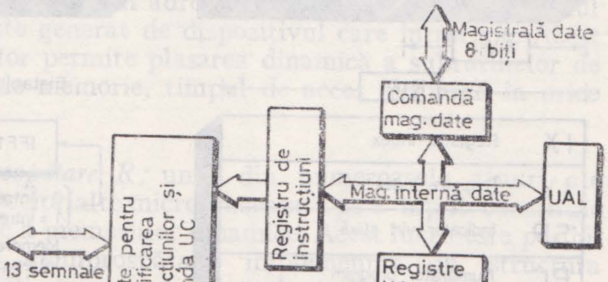


Fig. 1.1. Schema bloc a unității centrale UC-Z80

terne microprocesorului. Registrele interne, accesibile programatorului, sînt împărțite în două seturi de cîte șase registre generale ce pot fi folosite individual, ca registre de 8 biți, sau perechi, ca registre de 16 biți. Acumulatorul și registrul cu indicatorii de condiții sînt de asemenea, dublate.

UC-Z80 mai conține un indicator al vîrfului de stivă, *Stack Pointer*, un numărător de program, două registre index, un registru numărător pentru reîmprospătarea memoriei dinamice externe și un registru pentru memorarea întreruperilor. Unitatea centrală are nevoie pentru alimentare de o singură tensiune, +5V. Semnalele externe sînt decodificate și sincronizate, ceea ce permite interfațarea directă a microprocesorului cu memorii și circuite periferice standard.

1.1.1. REGISTRELE UNITĂȚII CENTRALE UC-Z80

În figura 1.2 sînt date cele trei grupuri de registre din cadrul UC-Z80. Primul grup este alcătuit din două seturi identice de registre de 8 biți, registrele principale (de exemplu A, B) și registrele secundare (de exemplu A′, B′). După cum se vede în figură, ambele seturi sînt compuse dintr-un acumulator, un registru cu indicatorii de condiții și șase registre generale de lucru. Transferul datelor între cele două seturi de registre se poate

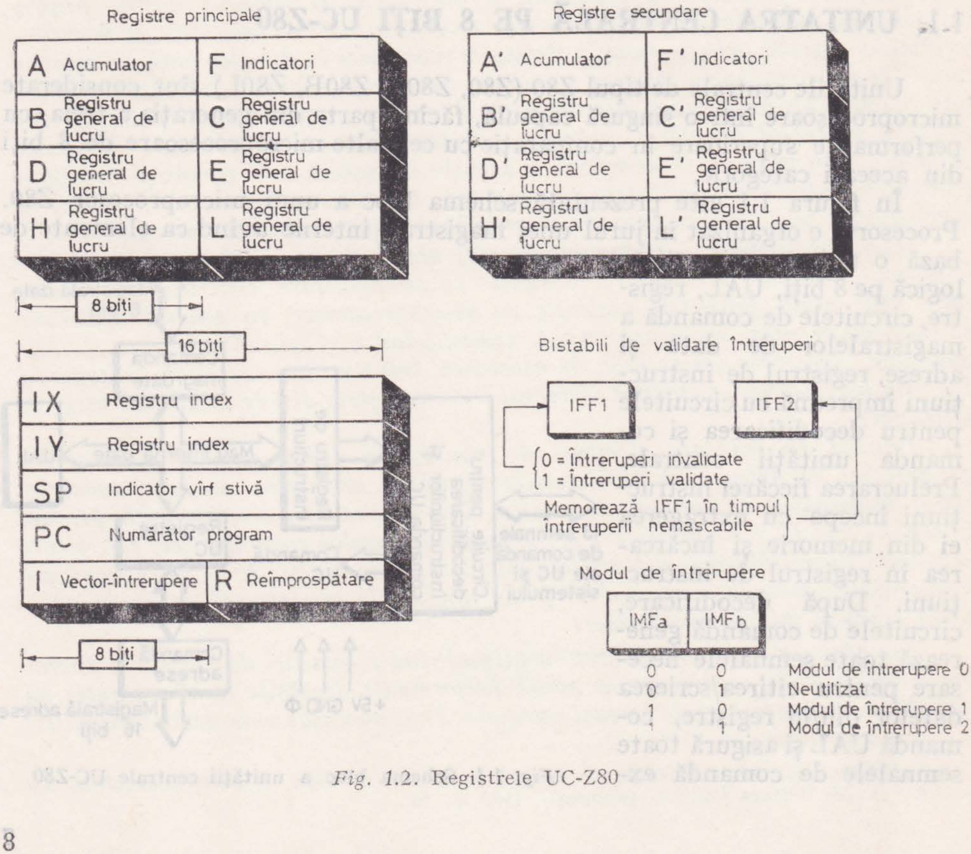


Fig. 1.2. Registrele UC-Z80

face cu ajutorul unor instrucţiuni *Exchange*. Existenţa celor două seturi identice de registre permite un răspuns mai rapid la întreruperi şi o implementare eficientă a unor tehnici de programare, cum este prelucrarea în *foreground/background*. De asemenea, se poate afirma că această structură de registre simplifică programarea, în special pentru sistemele cu multă memorie ROM şi puţină memorie RAM.

Al doilea grup de registre constă din şase registre cu funcţii specializate, specifice unităţilor centrale. Aceste registre sînt:

Numărătorul de program, PC, Program Counter; păstrează adresa de 16 biţi a instrucţiunii curente ce se extrage din memorie. PC este incrementat automat după ce conţinutul său se transferă în *buffer*-ul care comandă liniile de adrese externe. Atunci cînd în program apare un salt, se invalidează acţiunea circuitului intern de incrementare, în PC încărcîndu-se noua valoare a adresei de salt.

Indicatorul vîrfului de stivă, SP, Stack Pointer; păstrează adresa de 16 biţi a vîrfului stivei plasate oriunde în memoria RAM externă. Zona din memoria RAM utilizată ca stivă este organizată pe principiul „ultimul-intrat, primul-ieşit" *(LIFO — Last-In First-Out)*.

Registrele index, IX şi *IY*; cele două registre sînt destinate păstrării unor adrese de bază utilizabile în modul de adresare indexată. În această adresare, registrul index conţine adresa unei *zone* de memorie, baza. Deplasamentul, un număr cu semn exprimat în complement faţă de 2, este specificat prin intermediul unui octet suplimentar adăugat la instrucţiunile indexate.

Registrul pentru vectorul-întrerupere, I; UC-Z80 poate funcţiona într-un mod de tratare a întreruperilor cînd, ca răspuns la o cerere de întrerupere, se execută o instrucţiune de apel indirect la orice locaţie de memorie. Registrul I păstrează cei mai semnificativi 8 biţi ai adresei indirecte, în timp ce octetul cel mai puţin semnificativ este generat de dispozitivul care întrerupe. Acest mod de tratare a întreruperilor permite plasarea dinamică a subrutinelor de serviciu, oriunde în spaţiul de memorie, timpul de acces rămînînd în orice situaţie minim.

Registrul pentru reîmprospătare, R; unul din numeroasele atuuri ale microprocesorului Z80 în faţa celorlalte microprocesoare pe 8 biţi îl constituie posibilitatea conectării directe a memoriilor dinamice. Acest lucru este posibil datorită mecanismului de reîmprospătare implementat în structura UC-Z80. Registrul R de 7 biţi se incrementează după fiecare extragere de instrucţiune. Conţinutul lui se plasează pe porţiunea mai puţin semnificativă a magistralei de adrese, acţiune însoţită de generarea unui semnal de comandă a reîmprospătării. Reîmprospătarea se execută transparent din punct de vedere al utilizatorului, pe timpul cît unitatea centrală decodifică şi execută instrucţiunea extrasă, cînd magistrala e liberă.

Bistabilii de validare întreruperi, IFF1 şi IFF2; IFF1 semnalează starea de validare sau invalidare a întreruperilor în timp ce IFF2 stochează temporar

valoarea lui IFF1, pe timpul tratării întreruperii nemascabile. Poziționarea celor doi bistabili pe „1" şi pe „0" se face prin program cu ajutorul instrucțiunilor EI, respectiv DI.

Bistabilii de mod întrerupere, IMFa şi IMFb: codifică unul din cele trei moduri de lucru în întreruperi ale UC-Z80. Stabilirea modului de lucru se face prin program cu ajutorul instrucțiunilor IM 0, IM 1 sau IM 2.

1.1.2. SISTEMUL DE ÎNTRERUPERI

UC-Z80 acceptă două semnale de întrerupere: $\overline{NMI}$, întreruperea nemascabilă, şi $\overline{INT}$, întrerupere mascabilă validată selectiv prin program. La întreruperea nemascabilă Z80 răspunde într-un singur mod, în timp ce pentru întreruperea mascabilă există trei moduri de tratare. $\overline{NMI}$ este prioritară față de $\overline{INT}$.

La inițializare bistabilii IFF1 şi IFF2 sînt forțați pe zero, ceea ce echivalează cu invalidarea întreruperilor; în această stare microprocesorul nu acceptă întreruperi mascabile. Întreruperile se validează prin poziționarea bistabililor IFF1 şi IFF2 pe „1" cu ajutorul instrucțiunii EI. Orice întrerupere în aşteptare va putea fi servită numai după execuția instrucțiunii care urmează după EI. Întîrzierea de o instrucțiune este utilă atunci cînd după EI se execută o instrucțiune de revenire. În cazul în care UC-Z80 acceptă o întrerupere, IFF1 şi IFF2 sînt aduşi pe „0", inhibîndu-se astfel acceptarea unor alte întreruperi pînă la o nouă instrucțiune EI.

Aşa cum am menționat, destinația lui IFF2 este de a memora temporar starea lui IFF1 la apariția unei întreruperi nemascabile cînd, pentru prevenirea celorlalte întreruperi, IFF1 se forțează pe „0". Mai mult, la execuția unei instrucțiuni LD A, I sau LD A, R starea lui IFF2 este transcrisă în indicatorul de paritate, ceea ce permite testarea sau memorarea ei şi deci, implicit, refacerea prin program a valorii inițiale a lui IFF1. O altă cale, cea obişnuită, de a reface starea precedentă întreruperii nemascabile este prin execuția unei instrucțiuni de revenire din întreruperea nemascabilă, RETN.

Întreruperea nemascabilă nu poate fi invalidată prin program fiind acceptată în orice situație de UC-Z80. $\overline{NMI}$ se rezervă de obicei pentru evenimente prioritare, cum este căderea de tensiune. La apariția semnalului $\overline{NMI}$, dacă semnalul $\overline{BUSRQ}$ nu e activ, microprocesorul ignoră în ciclul de extragere următor codul instrucțiunii inițiind un restart la locația 0066H. La această adresă se găseşte, în general, secvența de serviciu a întreruperii nemascabile.

Z80 poate fi programat pentru a răspunde la întreruperile mascabile într-unul din modurile 0, 1 sau 2, memorate cu ajutorul bistabililor IMFa şi IMFb. Cele trei moduri sînt descrise în continuare.

Modul 0. Este compatibil cu procedurile de întrerupere ale microprocesorului 8080. În acest mod de întrerupere dispozitivul periferic poate plasa pe magistrala de date, în ciclul de tratare a întreruperii, orice instrucțiune. Deci, ideea procedurală este că instrucțiunea următoare nu se mai extrage din memorie, fiind furnizată de dispozitivul care întrerupe. În general, aceas-

ta e o instrucțiune *restart*, deoarece perifericul trebuie să asigure plasarea pe magistrala de date a unui singur octet. Instrucțiunile de restart realizează apeluri de subrutine plasabile la opt locații fixe în pagina zero de memorie. Desigur, dispozitivul care întrerupe poate genera orice cod de instrucțiune, de exemplu o instrucțiune CALL formată din trei octeți pentru apel la orice locație de memorie. Precizăm, de asemenea, că la inițializare UC-Z80 intră automat în modul 0.

Modul 1. Este foarte asemănător cu modul de răspuns la întreruperea nemascabilă. Diferența principală constă în faptul că se execută un restart la locația 0038H în loc de 0066H.

Modul 2. Este cel mai puternic mod de răspuns al microprocesorului Z80: cu un singur octet furnizat de dispozitivul care întrerupe se poate executa un *apel indirect* la orice locație de memorie.

Pentru a folosi acest mod de tratare a întreruperilor programatorul trebuie să scrie o tabelă cu adresele de început ale fiecărei rutine de serviciu. Tabela poate fi localizată în orice zonă a memoriei. La acceptarea unei întreruperi UC-Z80 formează un *pointer* de 16 biți cu ajutorul căruia ia din tabelă adresa rutinei de serviciu corespunzătoare dispozitivului care întrerupe. Cei mai semnificativi 8 biți ai *pointer*-ului sînt dați de conținutul registrului I încărcat în prealabil (la inițializare I=0). Cei mai puțin semnificativi 8 biți vor fi generați de periferic, cu observația că, deoarece ultimul bit trebuie să fie zero, sînt necesari de fapt numai 7. De aici rezultă că adresele de început ale subrutinelor de serviciu vor fi plasate în tabelă întotdeauna la adrese pare. În figura 1.3 se înfățișează procedura de tratare a întreruperii în modul 2: după ce dispozitivul periferic care întrerupe generează porțiunea cea mai puțin semnificativă a *pointer*-ului, UC-Z80 salvează automat în stivă numărătorul de program, obține din tabelă adresa de început a subrutinei de serviciu și efectuează un salt la această adresă.

Dispozitivele periferice din seria Z80 permit implementarea unui sistem de întreruperi care să lucreze în modul 2 asigurînd în acest scop generarea automată a vectorului de întrerupere pe timpul unui ciclu de achitare. De asemenea, menționăm că dispozitivele care întrerup pot fi conectate în lanț,

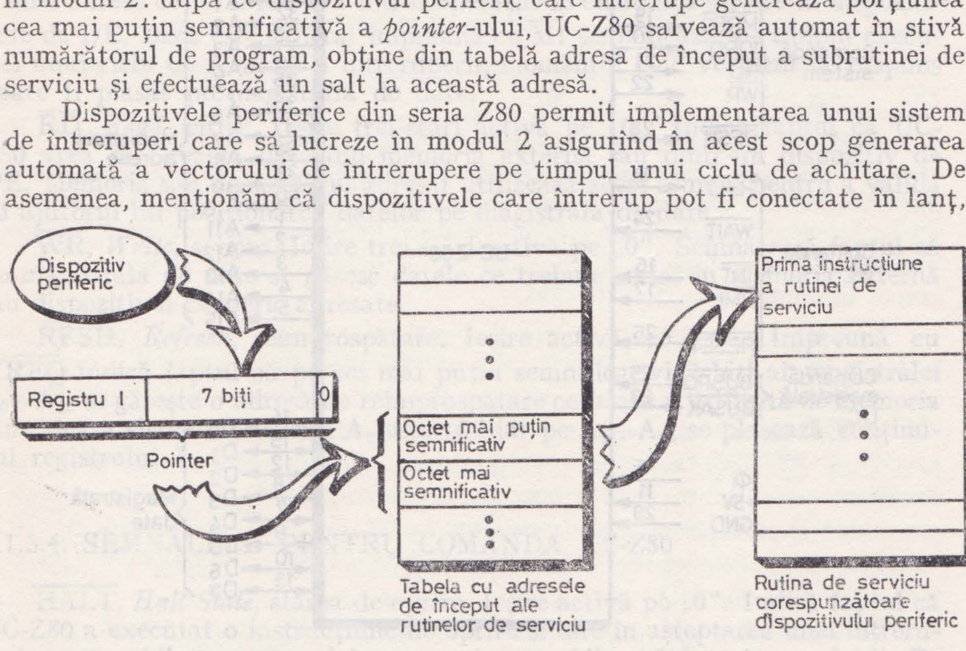


Fig. 1.3. Procedura de tratare a întreruperii în modul 2

prioritatea fiind determinată de poziția fizică în lanț. Fiecare circuit are o intrare de validare a întreruperilor, IEI, și o ieșire de validare a întreruperilor, IEO, către următorul dispozitiv. Primul dispozitiv din lanț, prioritar, trebuie să aibă intrarea IEI cablată la „1". Comanda semnalului IEO în funcție de IEI și de întreruperea locală este asigurată de toate circuitele periferice din seria Z80.

1.1.3. DESCRIEREA CONEXIUNILOR EXTERNE ALE UC-Z80

UC-Z80 este fabricat într-o capsulă de tip DIL, *Dual In-Line*, cu 40 de conexiuni externe. În figura 1.4 sînt indicate toate aceste conexiuni externe grupate pe funcțiuni: magistralele de adrese și de date, semnalele pentru comanda sistemului, unității centrale și magistralei UC-Z80, precum și ceasul și alimentarea microprocesorului. Vom descrie în continuare aceste semnale.

1.1.3.1. MAGISTRALA DE ADRESE

$A_0 \div A_{15}$, *Address Bus*. Ieșiri trei-stări active pe „1". Magistrala de adrese de 16 biți permite adresarea unei memorii externe de maximum 64 Kocteți și a intrării/ieșirii, I/E. Deoarece I/E se adresează cu cei mai puțin semnifica-

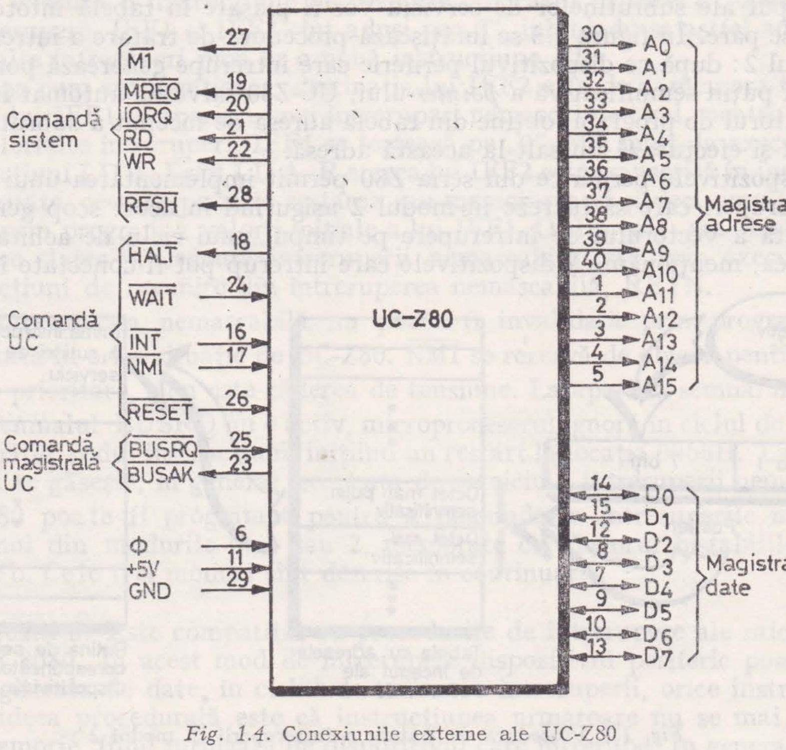


Fig. 1.4. Conexiunile externe ale UC-Z80

tivi 8 biți, se pot selecta 256 *port*-uri de intrare, respectiv 256 *port*-uri de ieșire. Pe timpul acțiunii de reîmprospătare a memoriilor dinamice cei mai puțin semnificativi 7 biți conțin adresa de reîmprospătare.

1.1.3.2. MAGISTRALA DE DATE

$D_0 \div D_7$, *Data Bus.* Intrări/ieșiri trei-stări active pe „1". Magistrala de date e utilizată pentru vehicularea bidirecțională a datelor între UC-Z80, memoria externă și dispozitivele de I/E.

1.1.3.3. SEMNALELE PENTRU COMANDA SISTEMULUI

$\overline{\text{M1}}$, *Machine Cycle One,* primul ciclu-mașină. Ieșire activă pe „0". Indică, atunci cînd apare împreună cu semnalul $\overline{\text{MREQ}}$, primul ciclu al execuției unei instrucțiuni, ciclul de extragere din memorie a codului-operație. Dacă apare în conjuncție cu $\overline{\text{IORQ}}$, semnifică începutul unui ciclu de achitare a întreruperii.

$\overline{\text{MREQ}}$, *Memory Request,* cerere la memorie. Ieșire trei-stări activă pe „0". Semnalează că pe magistrala de adrese s-a poziționat o adresă ce va fi utilizată într-o operație de citire sau scriere în memoria externă.

$\overline{\text{IORQ}}$, *Input/Output Request,* cerere de I/E. Ieșire trei-stări activă pe „0". Indică poziționarea pe octetul mai puțin semnificativ al magistralei $A_0 \div A_{15}$ a unei adrese ce va fi folosită într-o operație de citire sau scriere în dispozitivele de I/E. Dacă este generat împreună cu $\overline{\text{M1}}$ semnalează, pe timpul execuției unui ciclu de achitare a întreruperii, momentul cînd vectorul de răspuns poate fi plasat pe magistrala de date.

$\overline{\text{RD}}$, *Read,* citire. Ieșire trei-stări activă pe „0". Indică faptul că UC-Z80 vrea să citească date din memoria externă sau dintr-un dispozitiv de I/E. Memoria sau dispozitivul adresat utilizează acest semnal pentru a valida cu ajutorul lui poziționarea datelor pe magistrala de date.

$\overline{\text{WR}}$, *Write,* scriere. Ieșire trei-stări activă pe „0". Semnalează faptul că pe magistrala de date se găsesc datele ce trebuie scrise în memoria externă sau dispozitivul periferic adresate.

$\overline{\text{RFSH}}$, *Refresh,* reîmprospătare. Ieșire activă pe „0". Împreună cu $\overline{\text{MREQ}}$ indică faptul că pe cei mai puțin semnificativi 7 biți ai magistralei $A_0 \div A_{15}$ se găsește o adresă de reîmprospătare ce poate fi utilizată de memoria dinamică a sistemului. Bitul A_7 este „0" iar pe $A_8 \div A_{15}$ se plasează conținutul registrului I.

1.1.3.4. SEMNALELE PENTRU COMANDA UC-Z80

$\overline{\text{HALT}}$, *Halt State,* starea de oprire. Ieșire activă pe „0". Indică faptul că UC-Z80 a executat o instrucțiune de oprire și este în așteptarea unei întreruperi nemascabile sau a unei întreruperi mascabile validate în prealabil. Pe

timpul cît e oprită, UC-Z80 execută instrucțiuni NOP pentru a asigura funcția de reîmprospătare a memoriilor dinamice.

$\overline{\text{WAIT}}$, așteptare. Intrare activă pe „0". Indică microprocesorului că memoria sau dispozitivele de I/E adresate nu sînt gata pentru a efectua un transfer de date. UC-Z80 rămîne în starea de așteptare cît timp semnalul $\overline{\text{WAIT}}$ este activ. Pricizăm că pe durata cît semnalul $\overline{\text{WAIT}}$ este activ nu se generează $\overline{\text{RFSH}}$.

$\overline{\text{INT}}$, *Interrupt Request*, cerere de întrerupere. Intrare activă pe „0". Cerere de întrerupere generată de dispozitivele periferice. Cererea va fi achitată de UC-Z80 la sfîrșitul instrucțiunii în curs, cu condiția ca bistabilul de validare IFF1 să fi fost poziționat în prealabil și semnalul $\overline{\text{BUSRQ}}$ să nu fie activ. Achitarea în unul din cele trei moduri posibile se face cu un ciclu de achitare recunoscut prin aceea că se generează $\overline{\text{IORQ}}$ pe durata lui $\overline{\text{M1}}$. De obicei, la intrarea $\overline{\text{INT}}$ se conectează într-un SAU-cablat mai multe circuite, ceea ce necesită legarea unei rezistențe la +5V.

$\overline{\text{NMI}}$, *Non-Maskable Interrupt*, întrerupere nemascabilă. Intrare activă pe frontul negativ semnalînd o cerere de întrerupere nemascabilă. Prioritară față de $\overline{\text{INT}}$, întreruperea nemascabilă va fi întotdeauna recunoscută la sfîrșitul instrucțiunii curente, indiferent de starea bistabilului IFF1. $\overline{\text{NMI}}$ forțează o instrucțiune de restart la locația 0066H. Facem observațiile că durata instrucțiunii curente se poate mări prin intrarea în stări $\overline{\text{WAIT}}$ și că semnalul $\overline{\text{BUSRQ}}$ este prioritar față de $\overline{\text{NMI}}$.

$\overline{\text{RESET}}$, inițializare. Intrare activă pe „0". Activînd această intrare, se inițializează UC-Z80, adică:

– numărătorul de program se forțează pe zero;

– bistabilii IFF1 și IFF2 se pun pe „0", invalidîndu-se întreruperile;

– registrele I și R se fac 00H;

– se stabilește modul 0 de tratare a întreruperilor.

Pe timpul inițializării, magistralele de adrese și date trec în starea de impedanță mare, iar toate semnalele de comandă devin inactive. De asemenea, nu se generează semnale de reîmprospătare. Precizăm că pentru a se asigura inițializarea completă a microprocesorului intrarea $\overline{\text{RESET}}$ trebuie să fie activă minimum trei cicli de ceas.

$\overline{\text{BUSRQ}}$, *Bus Request*, cerere de magistrală. Intrare activă pe „0". Intrarea e folosită pentru a solicita din exteriorul microprocesorului preluarea magistralelor de adrese și date, precum și a ieșirilor trei-stări $\overline{\text{MREQ}}$, $\overline{\text{IORQ}}$, $\overline{\text{RD}}$ și $\overline{\text{WR}}$ astfel încît acestea să poată fi comandate de alte dispozitive. Semnalul $\overline{\text{BUSRQ}}$, prioritar față de $\overline{\text{NMI}}$, va fi întotdeauna achitat după terminarea ciclului-mașină în curs de execuție, UC-Z80 trecînd magistralele și ieșirile trei-stări pe starea de impedanță înaltă. Ca și $\overline{\text{INT}}$, la intrarea $\overline{\text{BUSRQ}}$ pot fi conectate mai multe circuite într-un SAU-cablat, ceea ce impune legarea unei rezistențe la +5V.

$\overline{\text{BUSAK}}$, *Bus Aknowledge*, achitare magistrală. Ieșire activă pe „0". Indică dispozitivului solicitant că magistralele de adrese și date, precum și

ieşirile trei-stări ale UC-Z80, au fost trecute pe starea de impedanţă înaltă, astfel încît acestea pot fi comandate din exterior. Facem observaţia că, pe durata cît ieşirea $\overline{BUSAK}$ e activă, nu se generează semnale de reîmprospătare.

1.1.4. DIAGRAME DE TIMP

Execuţia unei instrucţiuni a UC-Z80 este constituită dintr-o secvenţă specifică de operaţii ca, de exemplu, citire/scriere în memoria externă, citire/scriere într-un dispozitiv de I/E sau achitare de întrerupere. Fiecare dintre aceste operaţii de bază poate dura $4 \div 6$ perioade de ceas. Perioada ceasului Φ, este denumită ciclu sau stare $T(T_1, T_2...)$, iar operaţiile de bază, cicli-maşină M $(M_1, M_2, ...)$. Un ciclu-maşină M este format din mai multe stări T. Ciclii-maşină pot fi prelungiţi prin inserarea de stări WAIT. Execuţia unei instrucţiuni, un ciclu-instrucţiune, va însemna deci o succesiune de cîţiva cicli-maşină executaţi la rîndul lor în mai multe stări T (ca în figura 1.5).

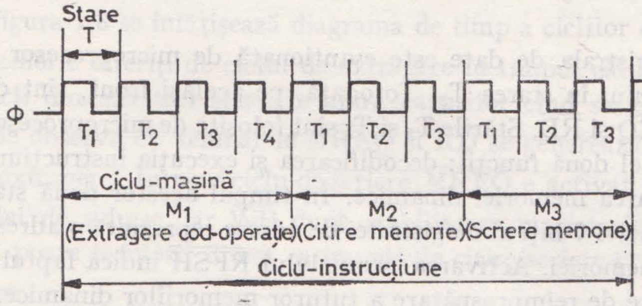


Fig. 1.5. Structura generală a unui ciclu-instrucţiune UC-Z80

Primul ciclu-maşină al oricărei instrucţiuni este un ciclu de extragere, *fetch*, a codului-operaţie al instrucţiunii ce urmează a fi executată. În următorii cicli UC-Z80 poate efectua transferuri de date cu memoria externă sau dispozitivele periferice. Vom prezenta în continuare desfăşurarea în timp a funcţionării UC-Z80 pentru operaţiile ce stau la baza execuţiei instrucţiunilor.

1.1.4.1. CICLUL DE EXTRAGERE A CODULUI-OPERAŢIE

Diagrama de timp e dată în figura 1.6. La începutul ciclului M_1 microprocesorul transferă conţinutul numărătorului de program PC pe magistrala de adrese. După aproximativ o jumătate de perioadă de ceas se activează semnalul $\overline{MREQ}$. În acest timp se consideră că liniile de adresă s-au stabilizat încît frontul negativ al lui $\overline{MREQ}$ poate fi utilizat direct pentru validarea memoriilor dinamice. O dată cu $\overline{MREQ}$ se activează şi comanda de citire $\overline{RD}$ pentru a indica plasarea datelor citite din memorie pe magistrala de date

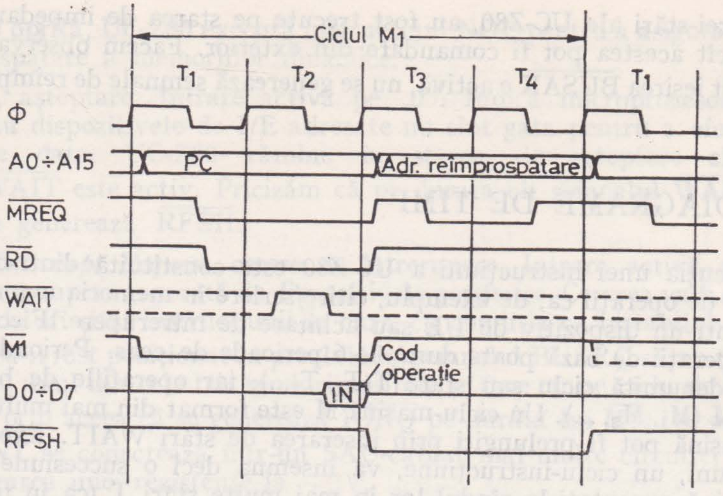


Fig. 1.6. Ciclul de extragere a codului-operaţie

a UC-Z80. Magistrala de date este eşantionată de microprocesor pe frontul pozitiv al ceasului în starea T_3. Totodată, pe acelaşi front, sînt dezactivate semnalele $\overline{MREQ}$ şi $\overline{RD}$. Stările T_3 şi T_4 sînt folosite de microprocesor pentru a realiza în paralel două funcţii: decodificarea şi execuţia instrucţiunii extrase, şi reîmprospătarea memoriei dinamice. În timpul acestor două stări cei mai puţin semnificativi 7 biţi ai magistralei de adrese reprezintă o adresă de reîmprospătare a memoriei. Activarea semnalului $\overline{RFSH}$ indică faptul că trebuie realizată o citire de reîmprospătare a tuturor memoriilor dinamice.

În figura 1.7 se prezintă un ciclu de extragere a codului-operaţie întîrziat datorită activării semnalului $\overline{WAIT}$. UC-Z80 eşantionează intrarea $\overline{WAIT}$,

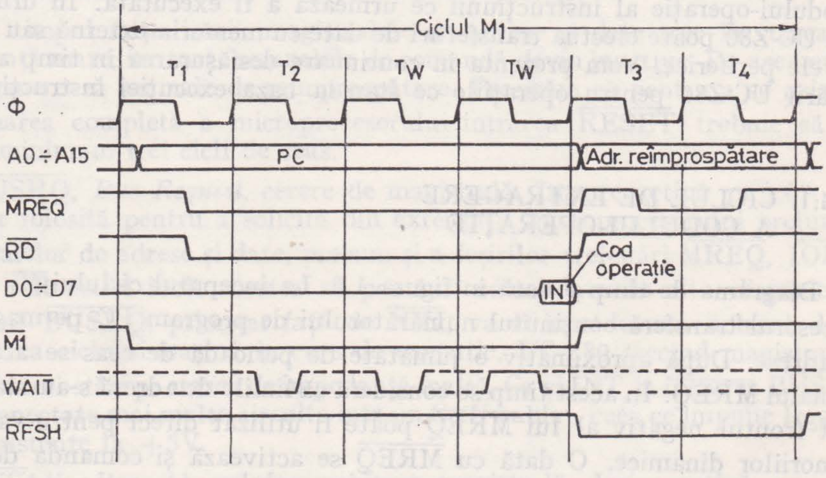


Fig. 1.7. Ciclul de extragere întîrziat cu stări WAIT

pe timpul stării T_2, cu frontul negativ al ceasului. Dacă este pe „0" se intră într-o stare de așteptare T_w. În timpul fiecărei stări T_w, pe frontul negativ al lui Φ, se investighează starea liniei $\overline{WAIT}$. Cît timp aceasta este „0", microprocesorul rămîne în starea T_w; la schimbarea ei pe „1" se trece în starea T_3, apoi T_4. Atragem din nou atenția că pe timpul stărilor T_w nu se generează semnale de reîmprospătare, pentru aceasta fiind nevoie de două stări succesive, așa cum sînt T_3 și T_4. Întîrzierea unei operații de citire din memoria externă prin activarea semnalului $\overline{WAIT}$ se folosește în cazurile cînd timpii de acces depășesc, aproximativ, durata unei perioade de ceas și, deci, stabilitatea datelor nu mai poate fi garantată la începutul stării T_3, pe frontul pozitiv al lui Φ.

1.1.4.2. CICLII DE CITIRE ȘI SCRIERE DIN/ÎN MEMORIA EXTERNĂ

În figura 1.8 se înfățișează diagrama de timp a ciclilor de citire și scriere din/în memorie diferiți de ciclul de extragere în timpul căruia se genera $\overline{M1}$. Acești cicli durează trei stări, în afara cazurilor cînd se activează intrarea $\overline{WAIT}$. Se observă că semnalele $\overline{MREQ}$ și $\overline{RD}$ se generează la fel ca într-un ciclu de extragere. Într-un ciclu de scriere, $\overline{MREQ}$ e activat după stabilizarea magistralei de adrese, iar $\overline{WR}$ după stabilizarea magistralei de date; acesta din urmă poate fi folosit direct ca impuls de citire/scriere pentru majoritatea memoriilor.

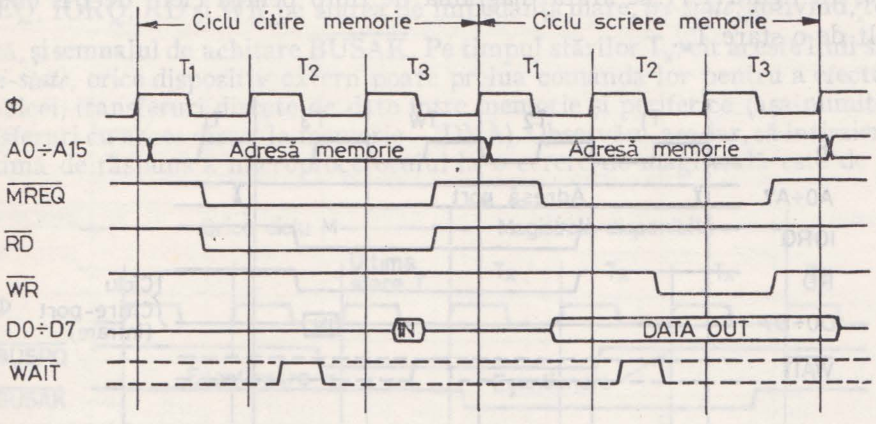


Fig. 1.8. Ciclii de citire și scriere din/în memoria externă

Ciclii de citire sau scriere întîrziați datorită activării semnalului $\overline{WAIT}$ pe durata frontului negativ al ceasului stării T_2 au o desfășurare în timp ca în figura 1.9.

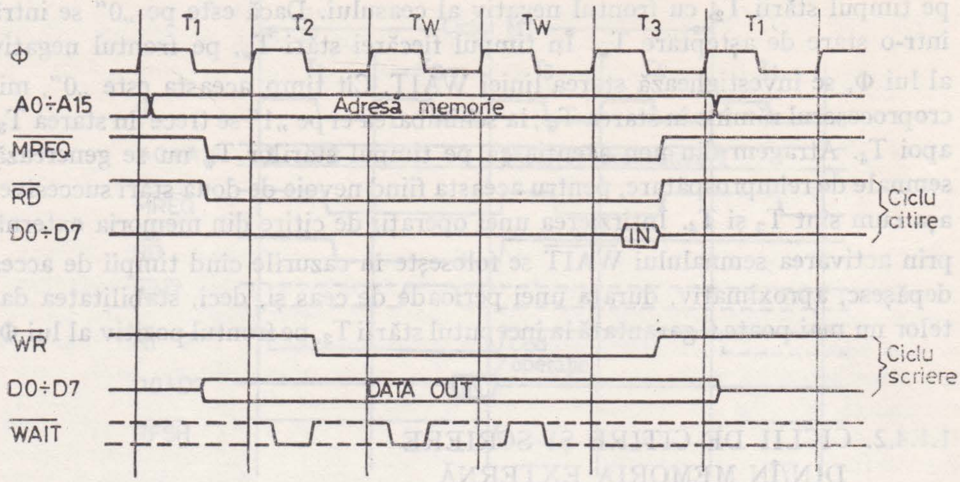


Fig. 1.9. Ciclii de citire și scriere întîrziați cu stări WAIT

1.1.4.3. CICLII DE INTRARE ȘI IEȘIRE

Figura 1.10 ilustrează modul de funcționare al microprocesorului Z80 în timpul operațiilor de intrare/ieșire. Pe durata acestor operații, UC-Z80 introduce în mod automat o stare T_w, după T_2, pentru a permite circuitului periferic adresat să-și decodifice adresa. Semnalul WAIT este testat pe timpul acestei stări T_w, în cazul în care e găsit activ menținîndu-se starea de așteptare. În figura 1.11 se arată diagrama de timp pentru ciclii de I/E cu mai mult de o stare T_w.

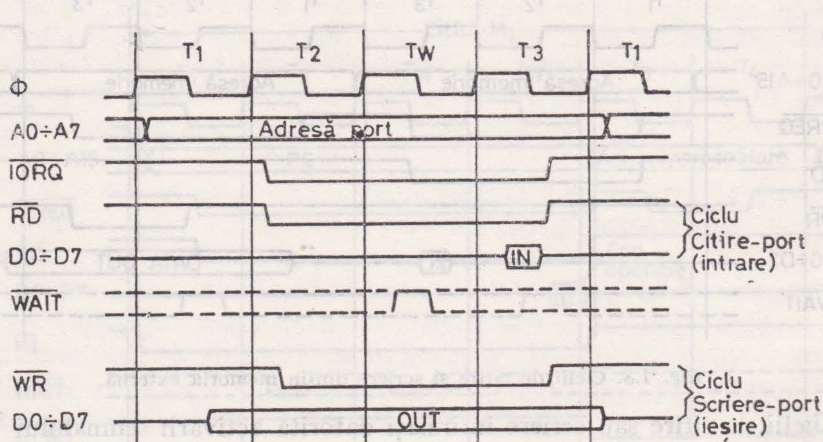


Fig. 1.10. Ciclii de intrare și ieșire

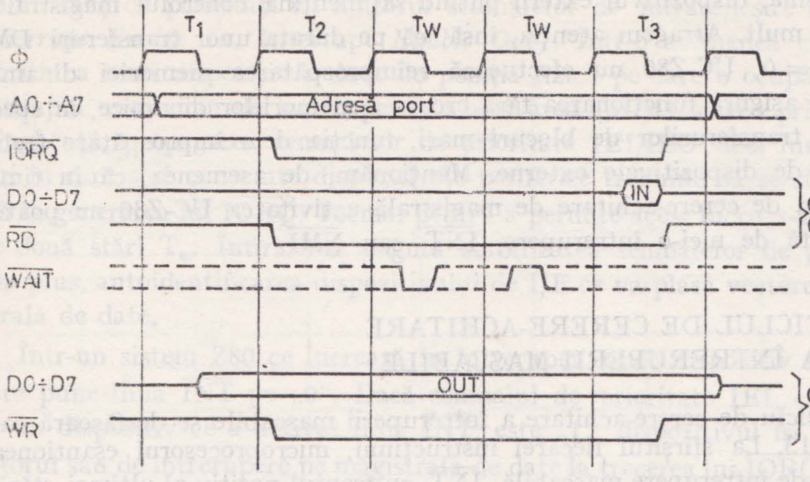


Fig. 1.11. Ciclii de intrare și ieșire întîrziați cu stări WAIT

1.1.4.4. CICLUL DE CERERE-ACHITARE DE MAGISTRALĂ

UC-Z80 eșantionează semnalul de cerere de magistrală, $\overline{BUSRQ}$, cu frontul pozitiv al ceasului, în ultima stare T a fiecărui ciclu mașină (vezi figura 1.12). Dacă $\overline{BUSRQ}$ este „0", pe frontul pozitiv al următoarei stări — T_x, microprocesorul va trece magistralele de adrese și date precum și ieșirile $\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$ și $\overline{WR}$ pe starea de impedanță mare, *tri-state*, activînd, totodată, și semnalul de achitare $\overline{BUSAK}$. Pe timpul stărilor T_x, cît aceste linii sînt în *tri-state,* orice dispozitiv extern poate prelua comanda lor pentru a efectua, de obicei, transferuri directe de date între memorie și periferice (așa-numitele transferuri cu acces direct la memorie — DMA). Observăm, așadar, că întîrzierea maximă de răspuns a microprocesorului la o cerere de magistrală este de un

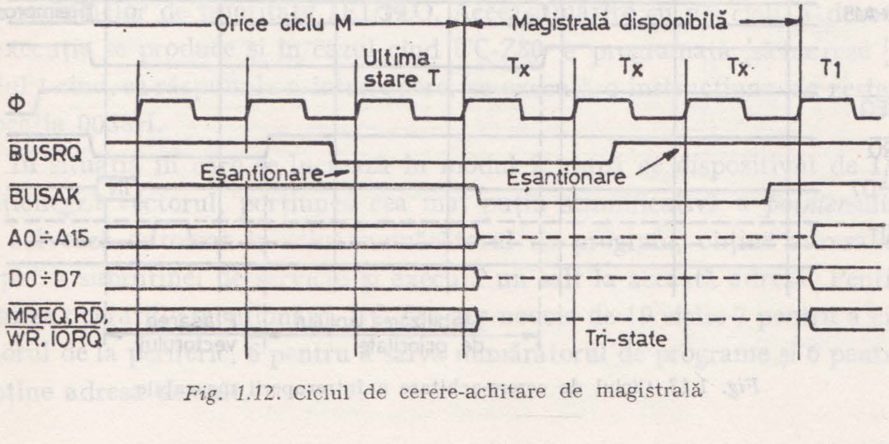


Fig. 1.12. Ciclul de cerere-achitare de magistrală

ciclu-mașină, dispozitivul extern putînd să mențină controlul magistralelor oricît de mult. Atragem atenția, însă, că pe durata unor transferuri DMA, $\overline{\text{BUSAK}} = 0$, UC-Z80 nu efectuează reîmprospătarea memoriei dinamice. Pentru a asigura funcționarea fără eroare a memoriilor dinamice, în special în cazul transferurilor de blocuri mari, funcția de reîmprospătare trebuie preluată de dispozitivele externe. Menționăm, de asemenea, că în timpul unui ciclu de cerere-achitare de magistrală activitatea UC-Z80 nu poate fi suspendată de nici-o întrerupere, $\overline{\text{INT}}$ sau $\overline{\text{NMI}}$.

1.1.4.5. CICLUL DE CERERE-ACHITARE A ÎNTRERUPERII MASCABILE

Un ciclu de cerere-achitare a întreruperii mascabile se desfășoară ca în figura 1.13. La sfîrșitul fiecărei instrucțiuni, microprocesorul eșantionează semnalul de întrerupere mascabilă, $\overline{\text{INT}}$, cu frontul pozitiv al ultimei stări T. Semnalul $\overline{\text{INT}}$ activ va fi acceptat numai dacă bistabilul intern de validare, IFF1, comandat prin software, este pus pe „1" și dacă semnalul de cerere de magistrală, $\overline{\text{BUSRQ}}$, nu e „0". Dacă întreruperea se acceptă, în continuare va fi declanșat un ciclu M_1 special: în timpul acestui ciclu, după poziționarea semnalului $\overline{\text{M1}}$ pe „0" se activează, în locul lui $\overline{\text{MREQ}}$ — cum se făcea într-un ciclu obișnuit de extragere —, $\overline{\text{IORQ}}$. Astfel se indică, într-un mod unic, dispozitivului periferic care întrerupe, să plaseze un vector de 8 biți pe magistrala de date.

În figura 1.13 se observă introducerea automată, de către microprocesor, a două stări de așteptare. Dispozitivele de I/E din seria Z80 pot fi conectate

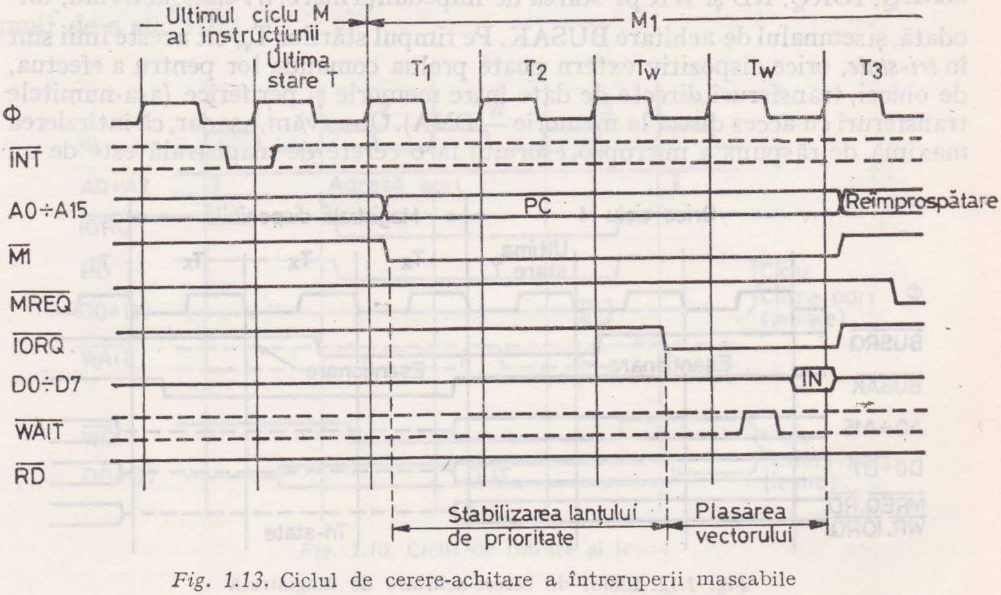


Fig. 1.13. Ciclul de cerere-achitare a întreruperii mascabile

într-un lanț de priorități cu ajutorul semnalelor de intrare/ieșire IEI/IEO *(Interrupt Enable In,* Interrupt *Enable Out).* Într-o asemenea conectare, prioritatea unui dispozitiv e dată de poziția fizică pe care o ocupă în lanț. Pe timpul unui ciclu de întrerupere, la activarea lui $\overline{M1}$ starea priorităților se îngheață, propagarea semnalelor de prioritate IEI/IEO de-a lungul lanțului în care sînt conectate dispozitivele periferice trebuind să se stabilizeze pînă la generarea lui $\overline{IORQ}$. Tocmai pentru a permite acest lucru s-au inserat cele două stări T_w. Întîrzierea asigură stabilizarea semnalelor de prioritate și, în plus, autoidentificarea dispozitivului de I/E ce va plasa vectorul pe magistrala de date.

Într-un sistem Z80 ce lucrează în întreruper orice dispozitiv solicitant poate pune linia $\overline{INT}$ pe „0". Dacă semnalul de prioritate IEI, care intră într-un dispozitiv ce a activat linia $\overline{INT}$, este „1", dispozitivul își va plasa vectorul său de întrerupere pe magistrala de date la trecerea lui $\overline{IORQ}$ pe „0". După aceasta, perifericul va elibera linia $\overline{INT}$ pentru a permite întreruperi de la dispozitivele prioritare. Dispozitivele cu prioritate scăzută sînt inhibate, deoarece semnalul de prioritate emis, IEO, este menținut pe „0". Semnalul IEO va fi pus pe „1" de către perifericul activ (cel care are IEI=1 și IEO=0), la decodificarea instrucțiunii de revenire RETI, adică la sfîrșitul subrutinei de serviciu. Poziționarea lui IEO pe „1" validează achitarea întreruperilor și pentru dispozitivele cu prioritate scăzută.

Microprocesorul Z80 poate fi programat să răspundă la întreruperea mascabilă în trei moduri posibile.

În modul 0, identic cu modul de funcționare al lui 8080, dispozitivul periferic este cel care generează instrucțiunea următoare. În acest caz ciclul de cerere-achitare a întreruperii este echivalent cu un ciclu de extragere, iar timpul necesar execuției instrucțiunii inserate de periferic e mai lung cu doi cicli decît atunci cînd codul-operație se citește din memorie. Cei doi cicli se datorează introducerii stărilor de așteptare, T_w, necesare stabilizării semnalelor de prioritate IEI/IEO. Aceeași mărire cu doi cicli a duratei de execuție se produce și în cazul cînd UC-Z80 e programată să lucreze în modul 1 cînd, ca răspuns la o întrerupere, se execută o instrucțiune de restart la locația 0038H.

În situația în care se lucrează în modul 2, după ce dispozitivul de I/E poziționează vectorul, porțiunea cea mai puțin semnificativă a *pointer*-ului, Z80 salvează automat în stivă numărătorul de program, obține adresa de început a subrutinei de serviciu și execută un salt la această adresă. Pentru a realiza operațiile menționate, UC-Z80 are nevoie de 19 cicli: 7 pentru a citi vectorul de la periferic, 6 pentru a salva numărătorul de programe și 6 pentru a obține adresa de salt.

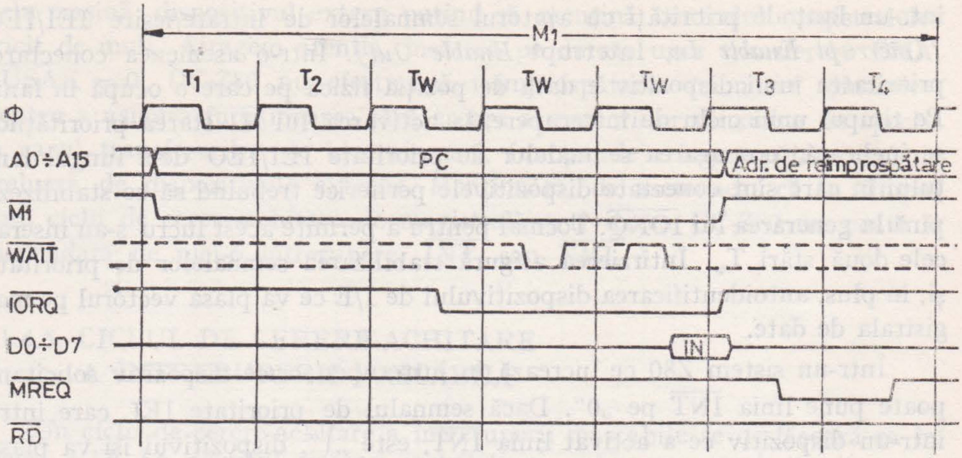


Fig. 1.14. Ciclul de cerere-achitare a întreruperii cu stări WAIT

În figura 1.14 se dă diagrama de timp a unui ciclu de cerere-achitare a întreruperii a cărui durată este mărită datorită activării de către periferic a liniei $\overline{WAIT}$.

1.1.4.6. CICLUL DE CERERE A ÎNTRERUPERII NEMASCABILE

Figura 1.15 descrie modul de funcționare a lui Z80 în cazul apariției unei întreruperi $\overline{NMI}$. $\overline{NMI}$ se eșantionează în același timp cu $\overline{INT}$, cu frontul pozitiv al ceasului, în ultima stare T a instrucțiunii în curs de execuție. $\overline{NMI}$ este însă prioritar față de $\overline{INT}$ și, de asemenea, nu poate fi mascat prin program. Un impuls pe linia $\overline{NMI}$ va conduce la poziționarea unui bistabil intern, acesta fiind de fapt testat la sfîrșitul fiecărei instrucțiuni. Răspunsul micro-

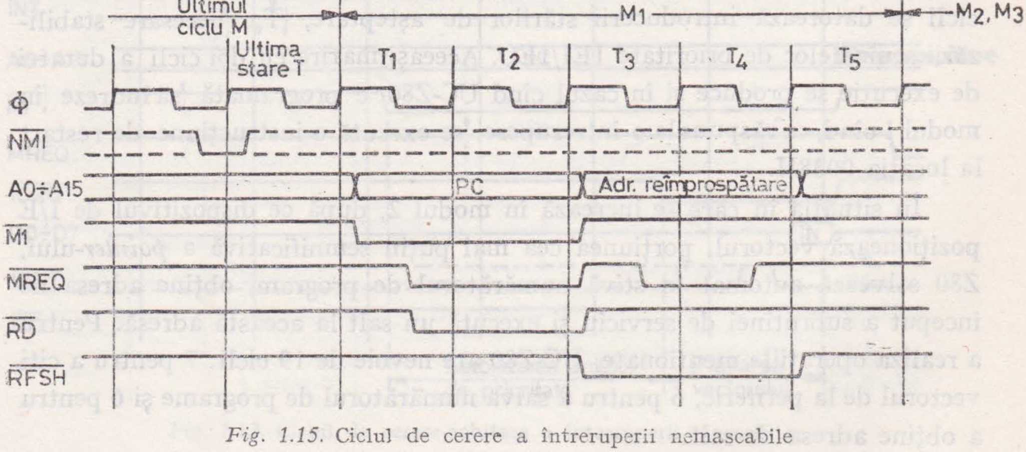


Fig. 1.15. Ciclul de cerere a întreruperii nemascabile

procesorului la $\overline{NMI}$ este asemănător cu un ciclu de citire din memorie. Diferența constă în aceea că, în situația de față, UC-Z80 ignoră codul plasat de memorie pe magistrala de date și execută o instrucțiune de restart la 0066H.

Pentru a înțelege mai bine relația între cele trei semnale, $\overline{BUSRQ}$, $\overline{NMI}$ și $\overline{INT}$, ce pot întrerupe fluxul normal al execuției instrucțiunilor, prezentăm în continuare organigrama din figura 1.16. Așa cum se vede, $\overline{INT}$ și $\overline{NMI}$ sînt luate în considerație la sfîrșitul instrucțiunii, în timp ce $\overline{BUSRQ}$

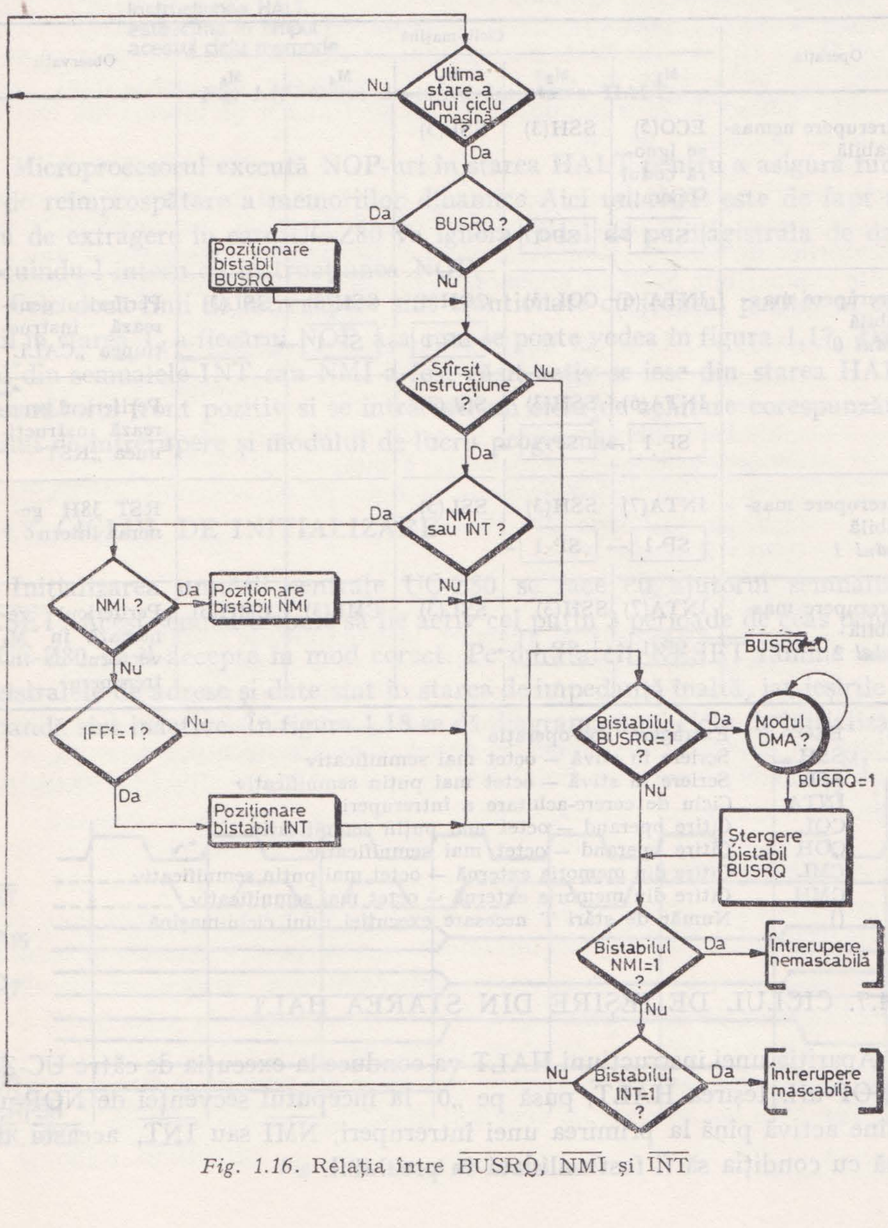


Fig. 1.16. Relația între $\overline{BUSRQ}$, $\overline{NMI}$ și $\overline{INT}$

la sfîrșitul unui ciclu-mașină. Ordinea de prioritate a acestor semnale, începînd cu cel prioritar, este: $\overline{\text{BUSRQ}}$. $\overline{\text{NMI}}$ și $\overline{\text{INT}}$. În timp ce UC-Z80 este în modul DMA, cu magistralele comandate de un periferic, nu se răspunde la $\overline{\text{NMI}}$ sau $\overline{\text{INT}}$.

Desfășurarea pe cicli-mașină a răspunsurilor UC-Z80 la întreruperi este dată în tabelul 1.1.

TABELUL 1.1. **Răspunsuri la întreruperi**

Operația	Ciclii-mașină					Observații
	M_1	M_2	M_3	M_4	M_5	
Întrerupere nemascabilă	ECO(5) se ignoră codul -operație SP-1 →	SSH(3) SP-1 →	SSL(3)			
Întrerupere mascabilă *Modul 0*	INTA (6)	COL(3)	COH(4) SP-1 →	SSH(3) SP-1 →	SSL(3)	Perifericul generează instrucțiunea „CALL"
	INTA(6) SP-1 →	SSH(3) SP-1 →	SSL(3)			Perifericul generează instrucțiunea „RST"
Întrerupere mascabilă *Modul 1*	INTA(7) SP-1 →	SSH(3) SP-1 →	SSL(3)			RST 38H generat intern
Întrerupere mascabilă *Modul 2*	INTA(7) SP-1 →	SSH(3) SP-1 →	SSL(3)	CML(3)	CMH(3)	Perifericul generează în M_1 vectorul de întrerupere

Notă:
- ECO — Extragere cod-operație
- SSH — Scriere în stivă — octet mai semnificativ
- SSL — Scriere în stivă — octet mai puțin semnificativ
- INTA — Ciclu de cerere-achitare a întreruperii
- COL — Citire operand — octet mai puțin semnificativ
- COH — Citire operand — octet mai semnificativ
- CML — Citire din memoria externă — octet mai puțin semnificativ
- CMH — Citire din memoria externă — octet mai semnificativ
- () — Număr de stări T necesare execuției unui ciclu-mașină

1.1.4.7. CICLUL DE IEȘIRE DIN STAREA HALT

Apariția unei instrucțiuni HALT va conduce la execuția de către UC-Z80 de NOP-uri. Ieșirea $\overline{\text{HALT}}$, pusă pe „0" la începutul secvenței de NOP-uri, rămîne activă pînă la primirea unei întreruperi, $\overline{\text{NMI}}$ sau $\overline{\text{INT}}$, aceasta din urmă cu condiția să fi fost validată în prealabil.

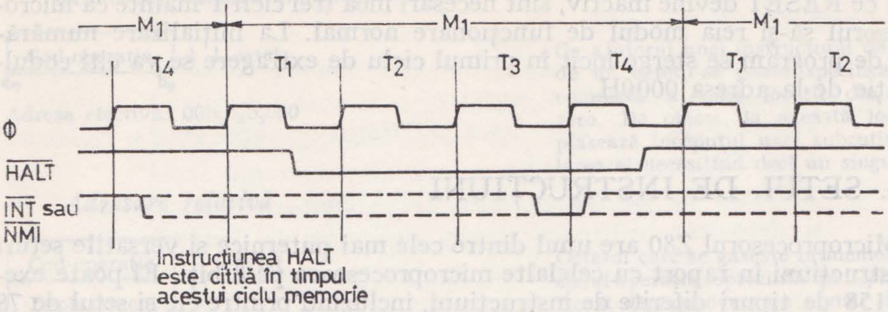


Fig. 1.17. Ciclul de ieşire din starea HALT

Microprocesorul execută NOP-uri în starea HALT pentru a asigura funcţia de reîmprospătare a memoriilor dinamice Aici un NOP este de fapt un ciclu de extragere în care UC-Z80 va ignora codul de pe magistrala de date înlocuindu-l intern cu instrucţiunea NOP.

Cele două linii de întrerupere sînt eşantionate cu frontul pozitiv al ceasului în starea T_4 a fiecărui NOP, aşa cum se poate vedea în figura 1.17. Dacă unul din semnalele $\overline{INT}$ sau $\overline{NMI}$ a fost găsit activ se iese din starea HALT pe următorul front pozitiv si se intră într-un ciclu de achitare corespunzător tipului de întrerupere şi modului de lucru programat.

1.1.4.8. CICLUL DE INIŢIALIZARE

Iniţializarea unităţii centrale UC-Z80 se face cu ajutorul semnalului $\overline{RESET}$. Acest semnal trebuie să fie activ cel puţin 3 perioade de ceas pentru ca UC-Z80 să îl accepte în mod corect. Pe durata cît $\overline{RESET}$ rămîne activ magistralele de adrese şi date sînt în starea de impedanţă înaltă, iar ieşirile de comandă sînt inactive. În figura 1.8 se dă diagrama unui ciclu de iniţializare.

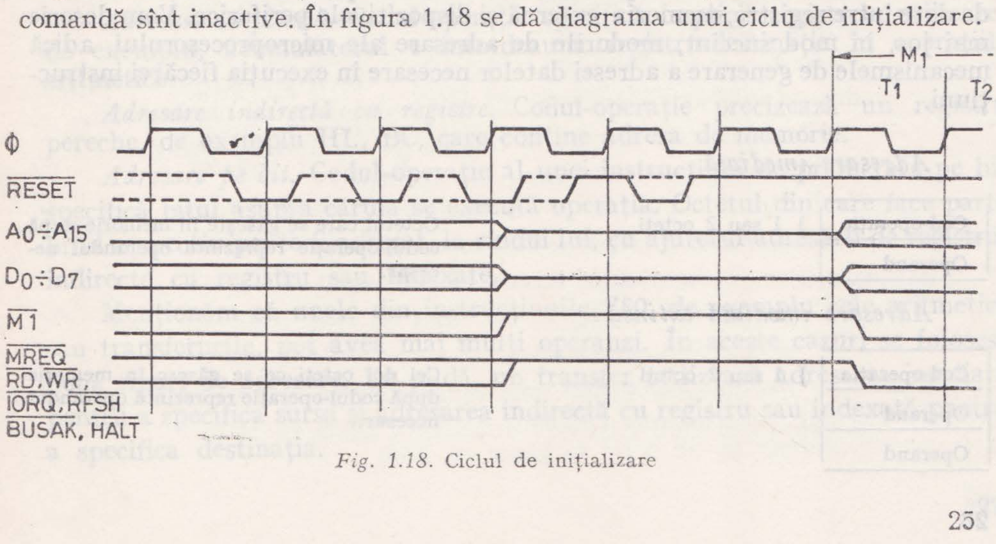


Fig. 1.18. Ciclul de iniţializare

După ce $\overline{RESET}$ devine inactiv, sînt necesari încă trei cicli T înainte ca microprocesorul să-şi reia modul de funcţionare normal. La iniţializare numărătorul de program se şterge încît în primul ciclu de extragere se va citi codul-operaţie de la adresa 0000H.

1.1.5. SETUL DE INSTRUCŢIUNI

Microprocesorul Z80 are unul dintre cele mai puternice şi versatile seturi de instrucţiuni în raport cu celelalte microprocesoare pe 8 biţi. El poate executa 158 de tipuri diferite de instrucţiuni, incluzînd printre ele şi setul de 78 instrucţiuni al lui 8080 (în cod obiect). Ca exemple de instrucţiuni foarte puternice putem da instrucţiunile speciale de transfer de blocuri sau instrucţiunile de prelucrare pe bit.

Setul de instrucţiuni Z80 va fi prezentat în continuare sub forma unor tabele sintetice, indicîndu-se: mnemonica instrucţiunii în limbaj de asamblare Z80, descrierea simbolică a operaţiei, codul-obiect al instrucţiunii, starea indicatorilor de condiţii, desfăşurarea pe cicli-maşină a instrucţiunii. Instrucţiunile compatibile 8080 au mnemonica în limbaj de asamblare 8080 trecută într-o coloană separată.

Instrucţiunile Z80 se pot împărţi în mai multe categorii:

— transferuri pe 8 biţi;
— transferuri pe 16 biţi;
— schimburi între registre, transfer de blocuri, căutări;
— operaţii aritmetice şi logice pe 8 biţi;
— operaţii aritmetice generale şi operaţii de comandă a UC-Z80;
— operaţii aritmetice pe 16 biţi;
— rotaţii şi deplasări;
— prelucrări pe bit;
— salturi;
— apeluri de subrutine, reveniri, instrucţiuni de restart;
— operaţii de I/E.

Modurile de adresare implementate în UC-Z80 permit un transfer eficient de date între registre, memoria externă şi dispozitivele periferice. Vom descrie mai jos, în mod succint, modurile de adresare ale microprocesorului, adică mecanismele de generare a adresei datelor necesare în execuţia fiecărei instrucţiuni.

Adresare imediată

Cod-operaţie	} 1 sau 2 octeţi	Octetul care se găseşte în memorie după codul-operaţie reprezintă operandul necesar.
Operand		

Adresare imediată extinsă

Cod-operaţie	} 1 sau 2 octeţi	Cei doi octeţi ce se găsesc în memorie după codul-operaţie reprezintă operandul necesar.
Operand		
Operand		

Adresare directă în pagina zero

Cod-operație	} 1 octet
b_7 … b_0	

Adresa efectivă: $00b_5b_4b_3000$

Cu ajutorul unei instrucțiuni de restart, de un octet, se poate specifica adresa completă a unei locații din pagina zero. De obicei, la această locație se plasează începutul unei subrutine, apelarea ei necesitînd deci un singur octet.

Adresare relativă

Cod-operație	} 1 octet
Deplasament	

Octetul care se găsește în memorie după codul-operație reprezintă un *deplasament*. Acest deplasament, un număr de 8 biți cu semn, scris în complement față de 2, se adună la adresa instrucțiunii următoare.

Adresare extinsă

Cod-operație	} 1 sau 2 octeți
Octetul mai puțin semnificativ al adresei de salt sau de operand	
Octetul mai semnificativ al adresei de salt sau de operand	

Cei doi octeți care urmează codului-operație constituie o adresă de salt sau o adresă de operand.

Adresare indexată

Cod-operație	} 2 octeți
Cod-operație	
Deplasament	

Octetul ce urmează codului-operație este un deplasament ce se adună la conținutul unuia dintre cele două registre index pentru a forma o adresă de memorie. Acest deplasament este, ca și în cazul adresării relative, un număr cu semn reprezentat în complement față de 2 $(-128, +127)$.

Adresare de registru. Codul-operație specifică explicit registrele utilizate de instrucțiune.

Adresare implicită. Codul-operație specifică implicit registrele utilizate: de exemplu, acumulatorul e întotdeauna registru-destinație în operațiile aritmetice.

Adresare indirectă cu registre. Codul-operație precizează un registru pereche, de exemplu HL, BC, care conține adresa de memorie.

Adresare pe bit. Codul-operație al unei instrucțiuni de prelucrare pe bit specifică bitul asupra căruia se execută operația. Octetul din care face parte bitul respectiv poate fi adresat, la rîndul lui, cu ajutorul adresării de registru, indirecte cu registru sau indexate.

Menționăm că unele din instrucțiunile Z80, de exemplu cele aritmetice sau transferurile, pot avea mai mulți operanzi. În aceste cazuri se folosesc două tipuri de adresare. De pildă, un transfer utilizează adresarea imediată pentru a specifica sursa și adresarea indirectă cu registru sau indexată pentru a specifica destinația.

În tabelele prezentate mai jos se vor folosi următoarele notații și simboluri pentru indicatorii de condiții:

S *Indicator de semn.* S=1 dacă cel mai semnificativ bit al rezultatului este „1", rezultatul e negativ.

Z *Indicator de zero.* Z=1 dacă rezultatul operației este o încărcare a lui 00H în acumulator.

P/V *Indicator de paritate/depășire.* După cum se vede, paritatea (P) și depășirea utilizează același indicator de condiție. Operațiile logice afectează indicatorul în funcție de paritatea rezultatului, iar cele aritmetice în funcție de depășire. Dacă P/V memorează paritatea atunci P/V=1 cînd rezultatul e par. În cazul în care P/V memorează depășirea atunci indicatorul e „1" cînd rezultatul produce depășire. Indicatorul de depășire, V, indică faptul că numărul în complement față de 2 din acumulator este eronat deoarece s-a depășit gama permisă: maximum +127, minimum −128. De exemplu:

$$
\begin{array}{r}
+119 = 0111\,0111 \\
+\ \ 9 = 0000\,1001 \\
\hline
C=0 \quad 1000\,0000
\end{array}
= -128 \text{ (greșit). Apare depășire.}
$$

În acest caz rezultatul e incorect: apare depășire iar indicatorul de transport rămîne „0". Situația va fi sesizată de indicatorul V care se va poziționa pe „1".

H *Indicator de transport auxiliar.* H=1 dacă operațiile de adunare sau scădere au produs un transport sau un împrumut din bitul 4 al acumulatorului. Indicatorul H nu poate fi testat prin program el fiind utilizat numai de instrucțiunea DAA.

N *Indicator de adunare/scădere.* N=1 dacă operația precedentă a fost scădere. De asemenea, nu poate fi testat prin program, fiind utilizat numai de DAA.

C *Indicator de transport.* C=1 dacă operația a produs un transport din cel mai semnificativ bit al operandului sau al rezultatului.

↕ Indicatorul este afectat în funcție de rezultatul operației.

• Indicatorul nu e modificat de operație.

0 Indicatorul e pus pe „0" de operație.

1 Indicatorul e pus pe „1" de operație.

× Valoarea indicatorului nu are importanță.

V P/V memorează depășirea.

P P/V memorează paritatea.

În scrierea mnemonicelor Z80 se vor întrebuința următoarele notații:

r,r' Unul din registrele UC-Z80: A, B, C, D, E, H, L.

n Expresie pe un octet în gama (0, 255).

(HL)	Conținutul memoriei la locația adresată de conținutul perechii de registre HL.
d	Expresie pe un octet în gama (−128, +127).
nn	Expresie pe doi octeți în gama (0, 65535).
(*nn*)	Conținutul memoriei la locația adresată de expresia *nn*.
dd	Unul din registrele duble: BC, DE, HL, SP.
qq	Unul din registrele duble: BC, DE, HL, AF.
s	Poate fi *r*, *n*, (HL), (IX+*d*) sau (IY+*d*).
m	Poate fi *r*, (HL), (IX+*d*) sau (IY+*d*).
ss	Unul din registrele duble BC, DE, HL, SP.
pp	Unul din registrele duble BC, DE, IX, SP.
rr	Unul din registrele duble BC, IY, SP, DE.
b	Expresie în gama (0, 7).
cc	Starea indicatorilor de condiții.
e	Expresie pe un octet în gama (−126, +129).

Pentru mnemonicele instrucțiunilor 8080 se mai folosesc următoarele simboluri:

rp	Unul din registrele pereche; B pentru BC, D pentru DE, H pentru HL și SP pentru indicatorul vîrfului de stivă.
PSW	Desemnează perechea de registre AF (acumulatorul și indicatorii de condiții).

Descrierea ciclilor-mașină se va face utilizînd următoarele prescurtări:

CM	Citire din memoria externă.
CMH	Citire din memoria externă — octet mai semnificativ.
CML	Citire din memoria externă — octet mai puțin semnificativ.
CO	Citire operand.
COH	Citire operand — octet mai semnificativ.
COL	Citire operand — octet mai puțin semnificativ.
CPO	Citire din *port* de I/E.
CSH	Citire din stivă — octet mai semnificativ.
CSL	Citire din stivă — octet mai puțin semnificativ.
ECO	Extragere cod-operație.
OI	Operație internă.
SM	Scriere în memoria externă.
SMH	Scriere în memoria externă — octet mai semnificativ.
SML	Scriere în memoria externă — octet mai puțin semnificativ.
SPO	Scriere în *port* de I/E.
SSH	Scriere în stivă — octet mai semnificativ.
SSL	Scriere în stivă — octet mai puțin semnificativ.
( )	Număr de stări T necesare execuției unui ciclu-mașină.

TABELUL 1.2. Transferuri pe 8 biţi

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect Binar	Codul-obiect Hexa	Indicatorii de condiţii S Z H P/V N C
LD *r,r'*	**MOV** *r,r'*	*r*←*r'*	01 *r* *r'*		. . × . × . . .
LD *r, n*	**MVI** *r, n*	*r*←*n*	00 *r* 110 ← *n* →		. . × . × . . .
LD *r*, (HL)	**MOV** *r*, M	*r*←(HL)	01 *r* 110		. . × . × . . .
LD *r*,(IX+*d*)		*r*←(IX+d)	11 011 101 01 *r* 101 ← *d* →	DD	. . × . × . . .
LD *r*,(IY+*d*)		*r*←(IY+*d*)	11 111 101 01 *r* 110 ← *d* →	FD	. . × . × . . .
LD (HL), *r*	**MOV** M, r	(HL)←*r*	01 110 *r*		. . × . × . . .
LD (IX+d), r		(IX+*d*)←*r*	11 011 101 01 110 *r* ← *d* →	DD	. . × . × . . .
LD (IY+*d*), *r*		(IY+*d*)←*r*	11 111 101 01 110 *r* ← *d* →	FD	. . × . × . . .
LD (HL), *n*	**MVI** M, n	(HL)←*n*	00 110 110 ← *n* →	36	. . × . × . . .
LD (IX+*d*), *n*		(IX+*d*)←*n*	11 011 101 00 110 110 ← *d* → ← *n* →	DD 36	. . × . × . . .
LD (IY+*d*),*n*		(IY+*d*)←*n*	11 111 101 00 110 110 ← *d* → ← *n* →	FD 36	. . × . × . . .
LD A, (BC)	**LDAX** B	A←(BC)	00 001 010	0A	. . × . × . . .
LD A, (DE)	**LDAX** D	A←(DE)	00 011 010	1A	. . × . × . . .
LD A, (nn)	**LDA** *nn*	A←(nn)	00 111 010 ← *n* → ← *n* →	3A	. . × . × . . .
LD (BC), A	**STAX** B	(BC)←A	00 000 010	02	. . × . × . . .
LD (DE), A	**STAX** D	(DE)←A	00 010 010	12	. . × . × . . .
LD (*nn*), A	**STA** *nn*	(*nn*)←A	00 110 010 ← *n* → ← *n* →	32	. . × . × . . .
LD A,I		A←I	11 101 101 01 010 111	ED 57	↕ ↕ × 0 × IFF 0 .
LD A,R		A←R	11 101 101 01 011 111	ED 5F	↕ ↕ × 0 × IFF 0 .
LD I,A		I←A	11 101 101 01 000 111	ED 47	. . × . × . . .
LD R,A		R←A	11 101 101 01 001 111	ED 4F	. . × . × . . .

Ciclii-mașină					Observații și note
M_1	M_2	M_3	M_4	M_5	
ECO(4)					r,r' — Registru 000 — B 001 — C 010 — D 011 — E 100 — H 101 — L 111 — A
ECO(4)	CO(3)				
ECO(4)	CM(3)				
ECO(4) + ECO(4)	CO(3)	OI(5)	CM(3)		
ECO(4) + ECO(4)	CO(3)	OI(5)	CM(3)		
ECO(4)	SM(3)				
ECO(4) + ECO(4)	CO(3)	OI(5)	SM(3)		
ECO(4) + ECO(4)	CO(3)	OI(5)	SM(3)		
ECO(4)	CO(3)	SM(3)			
ECO(4) + ECO(4)	CO(3)	CO(5)	SM(3)		
ECO(4) + ECO(4)	CO(3)	CO(5)	SM(3)		Nota 1
ECO(4)	CM(3)				
ECO(4)	CM(3)				
ECO(4)	COL(3)	COH(4)	CM(3)		
ECO(4)	SM(3)				
ECO(4)	SM(3)				
ECO(4)	COL(3)	C OH(3)	SM(3)		
ECO(4) + ECO(5)					Vezi § 1.1.3
ECO(4) + ECO(5)					
ECO(4) + ECO(5)					
ECO(4) + ECO(5)					

TABELUL 1.3. **Transferuri pe 16 biți**

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect Binar	Codul-obiect Hexa	Indicatorii de condiții S Z H P/V N C
LD *dd*, *nn*	**LXI** *rp*, *nn*	*dd*←*nn*	00 *dd*0 001 ← *n* → ← *n* →		. . × . × . . .
LD IX, *nn*		IX←*nn*	11 011 101 00 100 001 ← *n* → ← *n* →	DD 21	. . × . × . . .
LD IY, *nn*		IY←*nn*	11 111 101 00 100 001 ← *n* → ← *n* →	FD 21	. . × . × . . .
LD HL,(*nn*)	**LHLD** *nn*	H←(*nn*+1) L←(*nn*)	00 101 010 ← *n* → ← *n* →	2A	. . × . × . . .
LD *dd*,(*nn*)		dd_H←(*nn*+1) dd_L←(*nn*)	11 101 101 01 *dd*1 011 ← *n* → ← *n* →	ED	. . × . × . . .
LD IX,(*nn*)		IX_H←(*nn*+1) IX_L←(*nn*)	11 011 101 00 101 010 ← *n* → ← *n* →	DD 2A	. . × . × . . .
LD IY, (*nn*)		IY_H←(*nn*+1) IY_L←(*nn*)	11 111 101 00 101 010 ← *n* → ← *n* →	FD 2A	. . × . × . . .
LD (*nn*), HL	**SHLD** *nn*	(*nn*+1)←H (*nn*)←L	00 100 010 ← *n* → ← *n* →	22	. . × . × . . .
LD (*nn*), *dd*		(*nn*+1)←dd_H (*nn*)←dd_L	11 101 101 01 *dd*0 011 ← *n* → ← *n* →	ED	. . × . × . . .
LD (*nn*), IX		(*nn*+1)←IX_H (*nn*)←IX_L	11 011 101 00 100 010 ← *n* → ← *n* →	DD 22	. . × . × . . .
LD (*nn*), IY		(*nn*+1)←IY_H (*nn*)←IY_L	11 111 101 00 100 010 ← *n* → ← *n* →	FD 22	. . × . × . . .
LD SP, HL	**SPHL**	SP←HL	11 111 001	F9	. . × . × . . .
LD SP, IX		SP←IX	11 011 101 11 111 001	DD F9	. . × . × . . .

Ciclii-mașină					Observații și note
M_1	M_2	M_3	M_4	M_5	
ECO(4)	COL(3)	COH(3)			dd — Registre 00 — BC 01 — DE 10 — HL 11 — SP Nota 2
ECO(4) + ECO(4)	COL(3)	COH(3)			
ECO(4) + ECO(4)	COL(3)	COH(3)			
ECO(4)	COL(3)	COH(3)	CML(3)	CMH(3)	
ECO(4) + ECO(4)	COL(3)	COH(3)	CML(3)	CMH(3)	Indice H = mai semnificativ Indice L = mai puțin semnificativ
ECO(4) + ECO(4)	COL(3)	COH(3)	CML(3)	CMH(3)	
ECO(4) + ECO(4)	COL(3)	COH(3)	CML(3)	CMH(3)	
ECO(4)	COL(3)	COH(3)	SML(3)	SMH(3)	
ECO(4) + ECO(4)	COL(3)	COH(3)	SML(3)	SMH(3)	
ECO(4) + ECO(4)	COL(3)	COH(3)	SML(3)	SMH(3)	
ECO(4) + ECO(4)	COL(3)	COH(3)	SML(3)	SMH(3)	
ECO(6)					
ECO(4) + ECO(6)					

TABELUL 1.3 (continuare)

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect		Indicatorii de condiţii					
			Binar	Hexa	S	Z	H	P/V	N	C
LD SP, IY		SP←IY	11 111 101 11 111 001	FD F9	.	.	×	.	×	.	.	.
PUSH *qq*	PUSH *rp* PUSH PSW	(SP−2)←qq_L (SP−1)←qq_H SP←SP−2	11 *qq*0 101		.	.	×	.	×	.	.	.
PUSH IX		(SP−2)←IX_L (SP−1)←IX_H SP←SP−2	11 011 101 11 100 101	DD E5	.	.	×	.	×	.	.	.
PUSH IY		(SP−2)←IY_L (SP−1)←IY_H SP←SP−2	11 111 101 11 100 101	FD E5	.	.	×	.	×	.	.	.
POP *qq*	POP *rp* POP PSW	qq_H←(SP+1) qq_L←SP SP←SP+2	11 *qq*0 001		.	.	×	.	×	.	.	.
POP IX		IX_H←(SP+1) IX_L←SP SP←SP+2	11 011 101 11 100 001	DD E1	.	.	×	.	×	.	.	.
POP IY		IY_H←SP+1 IY_L←SP SP←SP+2	11 111 101 11 100 001	FD E1	.	.	×	.	×	.	.	.

TABELUL 1.4. Schimburi între registre, transfer de blocuri, căutări

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect		Indicatorii de condiţii					
			Binar	Hexa	S	Z	H	P/V	N	C
EX DE, HL	XCHG	DE↔HL	11 101 011	EB	.	.	×	.	×	.	.	.
EX AF,AF′		AF↔AF′	00 001 000	08	.	.	×	.	×	.	.	.
EXX		BC↔BC′ DE↔DE′ HL↔HL′	11 011 001	D9	.	.	×	.	×	.	.	.
EX (SP), HL	XTHL	H↔(SP+1) L↔(SP)	11 100 011	E3	.	.	×	.	×	.	.	.
EX (SP), IX		IX_H↔(SP+1) IX_L↔(SP)	11 011 101 11 100 011	DD E3	.	.	×	.	×	.	.	.

Ciclii-maşină M1	M2	M3	M4	M5	Observaţii şi note
ECO(4)+ ECO(6)					
ECO(5) SP−1 →	SSH(3) SP−1 →	SSL(3)			
ECO(4)+ ECO(5) SP−1 →	SSH(3) SP−1 →	SSL(3)			
ECO(4)+ ECO(5) SP−1 →	SSH(3) SP−1 →	SSL(3)			
ECO(4)	CSL(3) SP+1 →	CSH(3) SP+1 →			
ECO(4)+ ECO(4)	CSL(3) SP+1 →	CSH(3) SP+1 →			
ECO(4)+ ECO(4)	CSL(3) SP+1 →	CSH(3) SP+1 →			

Ciclii-maşină M1	M2	M3	M4	M5	Observaţii şi note
ECO(4)					
ECO(4)					
ECO(4)					
ECO(4)	CSL(3) SP+1 →	CSH(4)	SSH(3) SP−1 →	SSL(5)	
ECO(4)+ ECO(4)	CSL(3) SP+1 →	CSH(4)	SSH(3) SP−1 →	SSL(5)	

TABELUL 1.4 (continuare)

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect Binar	Codul-obiect Hexa	Indicatorii de condiții S Z H P/V N C
EX (SP), IY		IY_H↔(SP+1) IY_L↔(SP)	11 111 101 11 100 011	FD E3	. . × . × . .
LDI		(DE)←(HL) DE←DE+1 HL←HL+1 BC←BC−1	11 101 101 10 100 000	ED A0	. . × 0 × ↕ 0 .
LDIR		(DE)←(HL) DE←DE+1 HL←HL+1 BC←BC−1 se repetă pînă cînd BC=0	11 101 101 10 110 000	ED B0	. . × 0 × 0 0 .
LDD		(DE)←(HL) DE←DE−1 HL←HL−1 BC←BC−1	11 101 101 10 101 000	ED A8	. . × 0 × ↕ 0 .
LDDR		(DE)←(HL) DE←DE−1 HL←HL−1 BC←BC−1 se repetă pînă cînd BC=0	11 101 101 10 111 000	ED B8	. . × 0 × 0 0 .
CPI		A−(HL) HL←HL+1 BC←BC−1	11 101 101 10 100 001	ED A1	↕ ↕ × ↕ × ↕ 1 .
CPIR		A−(HL) HL←HL+1 BC←BC−1 se repetă pînă cînd A=(HL) sau BC=0	11 101 101 10 110 001	ED B1	↕ ↕ × ↕ × ↕ 1 .
CPD		A−(HL) HL←HL−1 BC←BC−1	11 101 101 10 101 001	ED A9	↕ ↕ × ↕ × ↕ 1 .
CPDR		(A −(HL) HL←HL−1 BC←BC−1 se repetă pînă cînd A=(HL) sau BC=0	11 101 101 10 111 001	ED B9	↕ ↕ × ↕ × ↕ 1 .

Ciclii-maşină					Observaţii şi note
M_1	M_2	M_3	M_4	M_5	
ECO(4)+ ECO(4)	CSL(3) SP+1 →	CSH(4)	SSH(3) SP−1 →	SSL(5)	
ECO(4)+ ECO(4)	CM(3)	SM(5)			Nota 3 Nota 4
ECO(4)+ ECO(4)	CM(3)	SM(5)	OI(5) numai cînd BC≠0		
ECO(4)+ ECO(4)	CM(3)	SM(5)			Nota 4
ECO(4)+ ECO(4)	CM(3)	SM(5)	OI(5) numai cînd BC≠0		
ECO(4)+ ECO(4)	CM(3)	OI(5)			Nota 4 Nota 5 Nota 6
ECO(4)+ ECO(4)	CM(3)	OI(5)	OI(5) numai cînd BC≠0		Nota 4 Nota 6
ECO(4)+ ECO(4)	CM(3)	OI(5)			Nota 4 Nota 6
ECO(4)+ ECO(4)	CM(3)	OI(5)	OI(5) numai cînd BC≠0		Nota 4 Nota 6

TABELUL 1.5. **Operații aritmetice și logice pe 8 biți**

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect Binar	Codul-obiect Hexa	Indicatorii de condiții S Z H P/V N C
ADD A, *r*	**ADD** *r*	A←A+*r*	10 000 *r*		↕ ↕ × ↕ × V 0 ↕
ADD A, *n*	**ADI** *n*	A←A+n	11 000 110 ← *n* →		↕ ↕ × ↕ × V 0 ↕
ADD A, (HL)	**ADD** M	A←A+(HL)	10 000 110		↕ ↕ × ↕ × V 0 ↕
ADD A,(IX+*d*)		A←A+(IX+*d*)	11 011 101 10 000 110 ← *d* →	DD	↕ ↕ × ↕ × V 0 ↕
ADD A,(IY+*d*)		A←A+(IY+*d*)	11 111 101 10 000 110 ← *d* →	FD	↕ ↕ × ↕ × V 0 ↕
ADC A, s	**ADC** *r* **ADC** M **ACI** *n*	A←A+*s*+CY	001		↕ ↕ × ↕ × V 0 ↕
SUB *s*	**SUB** *r* **SUB** M **SUI** *n*	A←A−*s*	010		↕ ↕ × ↕ × V 0 ↕
SBC A, *s*	**SBB** *r* **SBB** M **SBI** *n*	A←A−*s*−CY	011		↕ ↕ × ↕ × V 0 ↕
AND *s*	**ANA** *r* **ANA** M **ANI** *n*	A←A∧*s*	100		↕ ↕ × 1 × P 0 0
OR *s*	**ORA** *r* **ORA** M **ORI** *n*	A←A∨*s*	110		↕ ↕ × 0 × P 0 0
XOR *s*	**XRA** *r* **XRA** M **XRI** *n*	A←A⊕*s*	101		↕ ↕ × 0 × P 0 0
CP *s*	**CMP** *r* **CMP** M **CPI** *n*	A−*s*	111		↕ ↕ × ↕ × V 1 ↕
INC *r*	**INR** *r*	*r*←*r*+1	00 *r* 100		↕ ↕ × ↕ × V 0 .
INC (HL)	**INR** M	(HL)←(HL)+1	00 110 100		↕ ↕ × ↕ × V 0
INC (IX+*d*)		(IX+d)←(IX+ +*d*)+1	11 011 101 00 110 100 ← *d* →	DD	↕ ↕ × ↕ × V 0 .

Ciclii-mașină					Observații și note
M_1	M_2	M_3	M_4	M_5	
ECO(4)					*r* — Registru: 000 B 001 C 010 D 011 E 100 H 101 L 111 A
ECO(4)	CO(3)				
ECO(4)	CM(3)				
ECO(4)+ ECO(4)	CO(3)	OI(5)	CM(3)		
ECO(4)+ ECO(4)	CO(3)	OI(5)	CM(3)		
La fel ca la ADD, funcție de *s*					*s* poate fi *r*, *n*, (HL), (IX+*d*) sau (IY+*d*), ca în cazul instrucțiunilor ADD. Pentru a obține codul-obiect se înlocuiesc biții 000 din codul-obiect al instrucțiunii ADD corespunzătoare cu biții indicați.
		— „ —			
		— „ —			
		— „ —			
		— „ —			
		— „ —			
		— „ —			
ECO(4)					
ECO(4)	CM(4)	SM(3)			
ECO(4)+ ECO(4)	CO(3)	OI(5)	CM(4)	SM(3)	

TABELUL 1.5 (continuare)

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect Binar	Codul-obiect Hexa	Indicatorii de condiţii S Z H P/V N C
INC (IY+*d*)		(IY+*d*)←(IY+ +*d*)+1	11 111 101 00 110 |100| ← *d* →	FD	↕ ↕ × ↕ × V 0 .
DEC *m*	**DCR** *r* **DCR M**	*m*←*m*−1	|101|		↕ ↕ × ↕ × V 1 .

TABELUL 1.6. **Operaţii aritmetice generale şi operaţii de comandă a UC-Z80**

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect Binar	Codul-obiect Hexa	Indicatorii de condiţii S Z H P/V N C
DAA	**DAA**	Converteşte conţinutul acumulatorului într-un format BCD după adunări sau scăderi cu operanzi în format BCD	00 100 111	27	↕ ↕ × ↕ × P . ↕
CPL	**CMA**	A←$\overline{\text{A}}$	00 101 111	2F	. . × 1 × . 1 .
NEG		A←0−A	11 101 101 01 000 100	ED 44	↕ ↕ × ↕ × V 1 ↕
CCF	**CMC**	CY←$\overline{\text{CY}}$	00 111 111	3F	. . × × × . 0 ↕
SCF	**STC**	CY←1	00 110 111	37	. . × 0 × . 0 1
NOP	**NOP**		00 000 000	00	. . × . × . . .
HALT	**HLT**	Oprire UC-Z80	01 110 110	76	. . × . × . . .
DI	**DI**	IFF1,2←0	11 110 011	F3	. . × . × . . .
EI	**EI**	IFF1,2←1	11 111 011	FB	. . × . × . . .
IM 0		Stabilire mod întreruperi 0	11 101 101 01 000 110	ED 46	. . × . × . . .
IM 1		Stabilire mod întreruperi 1	11 101 101 01 010 110	ED 56	. . × . × . . .
IM 2		Stabilire mod întreruperi 2	11 101 101 01 011 110	ED 5E	. . × . × . . .

Ciclii-mașină					Observații și note
M_1	M_2	M_3	M_4	M_5	
ECO(4) + ECO(4)	CO(3)	OI(5)	CM(4)	SM(3)	
La fel ca la INC, funcție de *m*					*m* poare fi *r*, (HL), (IX+*d*) sau (IY+*d*), ca în cazul instrucțiunilor INC. Pentru a afla codul-obiect se înlocuiesc biții 100 cu 101.

Ciclii-mașină					Observații și note
M_1	M_2	M_3	M_4	M_5	
ECO(4)					Nota 7

ECO(4)					Complement față de 1. Nota 8
ECO(4) + ECO(4)					Complement față de 2. Nota 9
ECO(4)					
ECO(4)					
ECO(4)					
ECO(4)					
ECO(4)					Întreruperile nu sînt eșantionate la sfîrșitul instrucțiunilor EI și DI
ECO(4)					
ECO(4) + ECO(4)					
ECO(4) + ECO(4)					
ECO(4) + ECO(4)					

TABELUL 1.7. **Operații aritmetice pe 16 biți**

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect		Indicatorii de condiții
			Binar	Hexa	S Z H P/V N C
ADD HL, *ss*	**DAD** *rp*	HL←HL+*ss*	00 *ss* 1 001		. . × × × . 0 ↕
ADC HL, *ss*		HL←HL+*ss*+ +CY	11 101 101 01 *ss*1 010	ED	↕ ↕ × × × V 0 ↕
SBC HL, *ss*		HL←HL−*ss*− −CY	11 101 101 01 *ss*0 010	ED	↕ ↕ × × × V 1 ↕
ADD IX, *pp*		IX←IX+*pp*	11 011 101 01 *pp*1 001	DD	. . × × × . 0 ↕
ADD IY, *rr*		IY←IY+*rr*	11 111 101 00 *rr*1 001	FD	. . × × × . 0 ↕
INC *ss*	**INX** *rp*	*ss*←*ss*+1	00 *ss*0 011		. . × . × . . .
INC IX		IX←IX+1	11 011 101 00 100 011	DD 23	. . × . × . . .
INC IY		IY←IY+1	11 111 101 00 100 011	FD 23	. . × . × . . .
DEC *ss*	**DCX** *rp*	*ss*←*ss*−1	00 *ss*1 011		. . × . × . . .
DEC IX		IX←IX−1	11 011 101 00 101 011	DD 2B	. . × . × . . .
DEC IY		IY←IY−1	11 111 101 00 101 011	FD 2B	. . × . × . . .

TABELUL 1.8. **Rotații și deplasări**

Mnemonica Z80	Mnemonica 080	Descrierea simbolică	Codul-obiect		Indicatorii de condiții
			Binar	Hexa	S Z H P/V N C
RLCA	**RLC**	[CY]←[7←0]← A	00 000 111	07	. . × 0 × . 0 ↕
RLA	**RAL**	←[CY]←[7←0]← A	00 010 111	17	. . × 0 × . 0 ↕
RRCA	**RRC**	→[7→0]→[CY] A	00 001 111	0F	. . × 0 × . 0 ↕
RRA	**RAR**	→[7→0]→[CY]— A	00 011 111	1F	. . × 0 × . 0 ↕

Ciclii-mașină					Observații și note
M₁	M₂	M₃	M₄	M₅	
ECO(4)	OI(4)	OI(3)			*ss* — Registre 00 — BC 01 — DE 10 — HL 11 — SP
ECO(4)+ ECO(4)	OI(4)	OI(3)			
ECO(4)+ ECO(4)	OI(4)	OI(3)			
ECO(4)+ ECO(4)	OI(4)	OI(3)			*pp* — Registre 00 — BC 01 — DE 10 — IX 11 — SP
ECO(4)+ ECO(4)	OI(4)	OI(3)			
ECO(6)					*rr* — Registre 00 — BC 01 — DE 10 — IY 11 — SP
ECO(4)+ ECO(6)					
ECO(4)+ ECO(6)					
ECO(6)					
ECO(4)+ ECO(6)					
ECO(4)+ ECO(6)					

Ciclii-mașină					Observații și note
M₁	M₂	M₃	M₄	M₅	
ECO(4)					
ECO(4)					
ECO(4)					
ECO(4)					

TABELUL 1.8 (continuare)

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect		Indicatorii de condiții
			Binar	Hexa	S Z H P/V N C
RLC *r*			11 001 011 00 000 *r*	CB	↕ ↕ × 0 × P 0 ↕
RLC (HL)		CY ← 7←0 ← *r*	11 001 011 00 000 110	CB	↕ ↕ × 0 × P 0 ↕
RLC (IX+*d*)		(HL) (IX+*d*) (IY+*d*)	11 011 101 11 001 011 ← *d* → 00 000 110	DD CB	↕ ↕ × 0 × P 0 ↕
RLC (IY+*d*)			11 111 101 11 001 011 ← *d* → 00 000 110	FD CB	↕ ↕ × 0 × P 0 ↕
RL *m*		CY ← 7←0 ← *m*	010		↕ ↕ × 0 × P 0 ↕
RRC *m*		→ 7→0 → CY *m*	001		↕ ↕ × 0 × P 0 ↕
RR *m*		→ 7→0 → CY *m*	011		↕ ↕ × 0 × P 0 ↕
SLA *m*		CY ← 7←0 ← 0 *m*	100		↕ ↕ × 0 × P 0 ↕
SRA *m*		7→0 → CY ↑ *m*	101		↕ ↕ × 0 × P 0 ↕
SRL *m*		0 → 7→0 → CY *m*	111		↕ ↕ × 0 × P 0 ↕
RLD		7÷4 | 3÷0 7÷4 | 3÷0 A (HL)	11 101 101 01 101 111	ED 6F	↕ ↕ × 0 × P 0 .
RRD		7÷4 | 3÷0 7÷4 | 3÷0 A (HL)	11 101 101 01 100 111	ED 67	↕ ↕ × 0 × P 0 .

Ciclii-mașină					Observații și note
M_1	M_2	M_3	M_4	M_5	
ECO(4)+ ECO(4)					
ECO(4)+ ECO(4)					
ECO(4)+ ECO(4)	CO(3)	OI(5)	CM(4)	SM(3)	
ECO(4)+ ECO(4)	CO(3)	OI(5)	CM(4)	SM(3)	
La fel ca RLC, funcție de *m*					*m*=*r*, (HL), (IX+*d*), (IY+*d*) Pentru a obține codul-obiect se înlocuiesc biții 000 din codul-obiect al instrucțiunii RLC corespunzătoare cu biții indicați
		— „ —			
		— „ —			
		— „ —			
		— „ —			
		— „ —			
ECO(4)+ ECO(4)	CM(3)	OI(4)	SM(3)		
ECO(4)+ ECO(4)	CM(3)	OI(4)	SM(3)		

TABELUL 1.9. **Prelucrări pe bit**

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect Binar	Codul-obiect Hexa	Indicatorii de condiții S Z H P/V N C
BIT *b,r*		$Z \leftarrow \bar{r}_b$	11 001 011 01 *b* *r*	CB	× ↕ × 1 × × 0 .
BIT *b*, (HL)		$Z \leftarrow \overline{(HL)}_b$	11 001 011 01 *b* 110	CB	× ↕ × 1 × × 0 .
BIT *b*, (IX+*d*)		$Z \leftarrow \overline{(IX+d)}_b$	11 011 101 11 001 011 ← *d* → 01 *b* 110	DD CB	× ↕ × 1 × × 0 .
BIT *b*, (IY+*d*)		$Z \leftarrow \overline{(IY+d)}_b$	11 111 101 11 001 011 ← *d* → 01 *b* 110	FD CB	× ↕ × 1 × × 0 .
SET *b*, *r*		$r_b \leftarrow 1$	11 001 011 11 *b* *r*	CB	. . × . × . . .
SET *b*, (HL)		$(HL)_b \leftarrow 1$	11 001 011 11 *b* 110	CB	. . × . × . . .
SET *b*, (IX+*d*)		$(IX+d)_b \leftarrow 1$	11 011 101 11 001 011 ← *d* → 11 *b* 110	DD CB	. . × . × . . .
SET *b*, (IY+*d*)		$(IY+d)_b \leftarrow 1$	11 111 101 11 001 011 ← *d* → 11 *b* 110	FD CB	. . × . × . . .
RES *b*, *m*		$m_b \leftarrow 0$	10		. . × . × . . .

Ciclii-mașină					Observații și note
M_1	M_2	M_3	M_4	M_5	
ECO(4)+ ECO(4)					*r* – Registru 000 – B 001 – C 010 – D 011 – E 100 – H 101 – L 111 – A
ECO(4)+ ECO(4)	CM(4)				
ECO(4)+ ECO(4)	CO(3)	OI(5)	CM(4)		
ECO(4)+ ECO(4)	CO(3)	OI(5)	CM(4)		*b* – Bit testat 000 – 0 001 – 1 010 – 2 011 – 3 100 – 4 101 – 5 110 – 6 111 – 7
ECO(4)+ ECO(4)					
ECO(4)+ ECO(4)	CM(4)	SM(3)			
ECO(4)+ ECO(4)	CO(3)	OI(5)	CM(4)	SM(3)	
ECO(4)+ ECO(4)	CO(3)	OI(5)	CM(4)	SM(3)	
La fel ca SET, funcție de *m*					*m*=*r*, (HL), (IX+*d*), (IY+*d*) Pentru a obține codul-obiect se înlocuiesc biții |11| din codul-obiect al instrucțiunii SET corespunzătoare cu biții |10|.

TABELUL 1.10. Salturi

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect: Binar	Codul-obiect: Hexa	Indicatorii de condiţii: S Z H P/V N C
JP *nn*	**JMP** nn	PC←*nn*	11 000 011 ← *n* → ← *n* →	C3	. . × . × . . .
JP *cc*, *nn*	**Jcc** nn	Dacă *cc* adevărat PC←*nn*, altfel continuă	11 *cc* 010 ← *n* → ← *n* →		. . × . × . . .
JR *e*		PC←PC+*e*	00 011 000 ← *e*−2 →	18	. . × . × . . .
JR C, *e*		C=0 continuă C=1 PC←PC+ +*e*	00 111 000 ← *e*−2 →	38	. . × . × . . .
JR NC, *e*		C=1 continuă C=0 PC←PC+ +*e*	00 110 000 ← *e*−2 →	30	. . × . × . . .
JR Z, e		Z=0 continuă Z=1 PC←PC+ +*e*	00 101 000 ← *e*−2 →	28	. . × . × . . .
JR NZ, *e*		Z=1 continuă Z=0 PC←PC+ +*e*	00 100 000 ← *e*−2 →	20	. . × . × . . .
JP (HL)	**PCHL**	PC←HL	11 101 001	E9	. . × . × . . .
JP (IX)		PC←IX	11 011 101 11 101 001	DD E9	. . × . × . . .
JP (IY)		PC←IY	11 111 101 11 101 001	FD E9	. . × . × . . .
DJNZ, *e*		B←B−1 B=0 continuă B≠0 PC←PC+ +*e*	00 010 000 ← *e*−2 →	10	. . × . × . . .

Ciclii-mașină					Observații și note
M_1	M_2	M_3	M_4	M_5	
ECO(4)	COL(3)	COH(3)			*cc* — Condiția 000 NZ(non-zero) 001 Z(zero) 010 NC(non-carry) 011 C(carry) 100 PO(impar) 101 PE(par) 110 P(pozitiv) 111 M(negativ)
ECO(4)	COL(3)	COH(3)			
ECO(4)	CO(3)	OI(5)			*M_3 numai dacă se îndeplinește condiția (la salt).
ECO(4)	CO(3)	OI(5)*			*e* reprezintă un număr scris în complement față de 2 cuprins între −126 și +129
ECO(4)	CO(3)	OI(5)*			
ECO(4)	CO(3)	OI(5)*			
ECO(4)	CO(3)	OI(5)*			
ECO(4)					
ECO(4)+ ECO(4)					
ECO(4)+ ECO(4)					
ECO(4)	CO(3)	OI(5) dacă B≠0			

TABELUL 1.11. **Apeluri de subrutine, reveniri, instrucțiuni de restart**

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect Binar	Codul-obiect Hexa	Indicatorii de condiții S Z H P/V N C
CALL *nn*	CALL *nn*	$(SP-1) \leftarrow PC_H$ $(SP-2) \leftarrow PC_L$ $PC \leftarrow nn$	11 001 101 ← *n* → ← *n* →	CD	. . × . × . . .
CALL *cc*, *nn*	C*cc* *nn*	*cc*=0 continuă *cc*=1 CALL *nn*	11 *cc* 100 ← *n* → ← *n* →		. . × . × . . .
RET	RET	$PC_L \leftarrow (SP)$ $PC_H \leftarrow (SP+1)$	11 001 001	C9	. . × . × . . .
RET *cc*	R*cc*	*cc*=0 continuă *cc*=1 RET	11 *cc* 000		. . × . × . . .
RETI		Revenire din întrerupere	11 101 101 01 001 101	ED 4D	. . × . × . . .
RETN		Revenire din întrerupere nemascabilă	11 101 101 01 000 101	ED 45	. . × . × . . .
RST *p*	RST *t*	$(SP-1) \leftarrow PC_H$ $(SP-2) \leftarrow PC_L$ $PC_H \leftarrow 0$ $PC_L \leftarrow p$	11 *t* 111		. . × . × . . .

TABELUL 1.12. **Operații de intrare/ieșire**

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect Binar	Codul-obiect Hexa	Indicatorii de condiții S Z H P/V N C
IN A, (*n*)	IN *n*	$A \leftarrow (n)$	11 011 011 ← *n* →	DB	. . × . × . . .
IN *r*, (C)		$r \leftarrow (C)$	11 101 101 01 *r* 000	ED	↕ ↕ × ↕ × P 0 .
INI		$(HL) \leftarrow (C)$ $B \leftarrow B-1$ $HL \leftarrow HL+1$	11 101 101 10 100 010	ED A2	× ↕ × × × × ↕ ×

Ciclii-mașină					Observații și note
M_1	M_2	M_3	M_4	M_5	
ECO(4)	COL(3)	COH(4) | SP − 1 | →	SSH(3) | SP − 1 | →	SSL(3)	
ECO(4)	COL(3)	COH(3)			
ECO(4)	COL(3)	COH(4) | SP − 1 | →	SSH(3) | SP − 1 | →	SSL(3)	
ECO(4)	CSL(3) | SP + 1 | →	CSH(3) | SP + 1 | →			
ECO(5)	CSL(3) dacă *cc* = 1 | SP + 1 | →	CSH(3) dacă *cc* = 1 | SP + 1 | →			
ECO(4) + ECO(4)	CSL(3) | SP + 1 | →	CSH(3) | SP + 1 | →			
ECO(4) + ECO(4)	CSL(3) | SP + 1 | →	CSH(3) | SP + 1 | →			
ECO(5) | SP − 1 | →	SSH(3) | SP − 1 | →	SSL(3)			

	cc	Condiția
cc = 0	000	NZ(non-zero)
	001	Z(zero)
	010	NC(non-carry)
	011	C(carry)
	100	PO(impar)
	101	PE(par)
	110	P(pozitiv)
cc = 1	111	M(negativ)

t	*p*
000	00H
001	08H
010	10H
011	18H
100	20H
101	28H
110	30H
111	38H

Ciclii-mașină					Observații și note
M_1	M_2	M_3	M_4	M_5	
ECO(4)	CO(3)	CPO(4)			$\{A_0 \div A_7 \leftarrow n$ $\{A_8 \div A_{15} \leftarrow A$
ECO(4) + ECO(4)	CPO(4)				$\{A_0 \div A_7 \leftarrow C$ Nota 7 $\{A_8 \div A_{15} \rightarrow B$
ECO(4) + ECO(5)	CPO(4)	SM(3)			$\{A_0 \div A_7 \leftarrow C$ $\{A_8 \div A_{15} \leftarrow B$ Nota 8

TABELUL 1.12 (continuare)

Mnemonica Z80	Mnemonica 8080	Descrierea simbolică	Codul-obiect		Indicatorii de condiții
			Binar	Hexa	S Z H P/V N C
INIR		(HL)←(C) B←B−1 HL←HL+1 se repetă pînă cînd B=0	11 101 101 10 110 010	ED B2	× 1 × × × × ↕ ×
IND		(HL)←(C) B←B−1 HL←HL−1	11 101 101 10 101 010	ED AA	× ↕ × × × × ↕ ×
INDR		(HL)←(C) B←B−1 HL←HL−1 se repetă pînă cînd B=0	11 101 101 10 111 010	ED BA	× 1 × × × × ↕ ×
OUT (*n*), A	**OUT** *n*	(*n*)←A	11 010 011 ← *n* →	D3	. . × . × . ↕ ×
OUT (C), *r*		(C)←*r*	11 101 101 01 *r* 001	ED	. . × . × . ↕ ×
OUTI		(C)←(HL) B←B−1 HL←HL+1	11 101 101 10 100 011	ED A3	× ↕ × × × × ↕ ×
OTIR		(C)←(HL) B←B−1 HL←HL+1 se repetă pînă cînd B=0	11 101 101 10 110 011	ED B3	× 1 × × × × ↕ ×
OUTD		(C)←(HL) B←B−1 HL←HL−1	11 101 101 10 101 011	ED AB	× ↕ × × × × ↕ ×
OTDR		(C)←(HL) B←B−1 HL←HL−1 se repetă pînă cînd B=0	11 101 101 10 111 011	ED BB	× 1 × × × × ↕ ×

Ciclii-mașină					
M_1	M_2	M_3	M_4	M_5	Observații și note
ECO(4)+ ECO(5)	CPO(4)	SM(3)	OI(5) dacă $B \neq 0$		$A_0 \div A_7 \leftarrow C$ $A_8 \div A_{15} \leftarrow B$
ECO(4)+ ECO(5)	CPO(4)	SM(3)			$A_0 \div A_7 \leftarrow C$ $A_8 \div A_{15} \leftarrow B$ Nota 8
ECO(4)+ ECO(5)	CPO(4)	SM(3)	OI(5) dacă $B \neq 0$		$A_0 \div A_7 \leftarrow C$ $A_8 \div A_{15} \leftarrow B$
ECO(4)	CO(3)	SPO(4)			$A_0 \div A_7 \leftarrow n$ $A_8 \div A_{15} \leftarrow A$
ECO(4)+ ECO(4)	SPO(4)				$A_0 \div A_7 \leftarrow C$ Nota 7 $A_8 \div A_{15} \leftarrow B$
ECO(4)+ ECO(5)	CM(3)	SPO(4)			$A_0 \div A_7 \leftarrow C$ $A_8 \div A_{15} \leftarrow B$ Nota 8
ECO(4)+ ECO(5)	CM(3)	SPO(4)	OI(5) dacă $B \neq 0$		$A_0 \div A_7 \leftarrow C$ $A_8 \div A_{15} \leftarrow B$
ECO(4)+ ECO(5)	CM(3)	SPO(4)	OI(5)		$A_0 \div A_7 \leftarrow C$ $A_8 \div A_{15} \leftarrow B$ Nota 8
ECO(4)+ ECO(5)	CM(3)	SPO(4)	OI(5) dacă $B \neq 0$		$A_0 \div A_7 \leftarrow C$ $A_8 \div A_{15} \leftarrow B$

NOTE:

1. Exemplu de transfer într-o locație de memorie folosind adresarea indexată pentru destinație și adresarea imediată pentru sursă:

LD (IX−31), 44H

Instrucțiunea va apărea în memorie scrisă în următoarea ordine:

Adresa A	DD	} Cod-operație
A+1	36	
A+2	E1	} Deplasament în complement față de 2
A+3	44	} Operand

2. Exemplu de instrucțiune de transfer pe 16 biți utilizînd adresarea imediată pentru sursă și adresarea de registru pentru destinație:

LD BC, 176AH

Instrucțiunea va apărea în memorie scrisă astfel:

Adresa A	01	} Cod-operație
	6A	} Operand mai puțin semnificativ → C
	17	} Operand mai semnificativ → B

3. Instrucțiunile pentru transfer de blocuri de date, LDI, LDIR, LDD și LDDR, utilizează toate, în același mod, următoarele registre:

HL, pentru a adresa locația-sursă,

DE, pentru a adresa locația-destinație și

BC, ca numărător de octeți.

Pentru folosirea corectă a celor patru instrucțiuni de transfer trebuie ca registrele HL, DE și BC să fie inițializate, în prealabil, conform operației ce se dorește a fi executată.

LDI, *Load and Increment*, încărcare și incrementare, transferă un octet din locația adresată de HL în locația adresată cu DE. Registrele HL și DE se incrementează pentru a fi pregătite să adreseze următoarele locații, iar numărătorul de octeți BC se decrementează. Instrucțiunea e utilă atunci cînd între transferul a doi octeți sînt necesare prelucrări suplimentare. Dacă aceste prelucrări nu sînt necesare se poate folosi instrucțiunea LDIR, *Load, Increment and Repeat*, încărcare, incrementare și repetare, care repetă transferul pînă cînd numărătorul de octeți devine zero. Menționăm că lungimea maximă a unui bloc poate fi de 64 Kocteți și că zonele de transfer se pot suprapune.

Instrucțiunile LDD și LDDR sînt asemănătoare cu LDI și LDIR, diferența constînd numai în aceea că registrele de adresare, HL și DE, sînt *decrementate* după fiecare transfer. Deci LDI și LDIR inițiază transferul cu adresa de început a blocului, în timp ce LDD și LDDR cu adresa de sfîrșit.

Exemplu:

Înainte			După
HL : 3333H			HL : 3334H
(3333H): 55H			(3333H): 55H
DE : 7777H	Executînd instrucțiunea LDI se obține următoarea situație:		DE : 7778H
(7777H): 99H			(7777H): 55H
BC : 0BH			BC : 0AH

4. Indicatorul P/V se pune pe „0" dacă BC−1=0. În celelalte cazuri P/V rămîne „1".

5. Instrucţiunile de căutare, CPI, CPIR, CPD şi CPDR, compară acumulatorul cu conţinutul locaţiei adresate de HL. Rezultatul comparaţiei se memorează în indicatorul de condiţii Z. Ca şi în cazul transferurilor, CPI, *Compare and Increment*, după ce compară acumulatorul cu (HL), incrementează registrul HL şi decrementează numărătorul de octeţi. Dacă nu sînt necesare prelucrări suplimentare între două comparări de octeţi succesivi, se poate utiliza instrucţiunea CPIR, *Compare, Increment and Repeat*, care repetă comparaţia fie pînă cînd A=(HL), fie pînă cînd BC=0.

CPD şi CPDR sînt asemănătoare cu CPI şi CPIR diferenţa constînd numai în faptul că registrul HL este *decrementat* după fiecare comparaţie.

Exemplu:

HL	1111H	Executînd instrucţiunea CPIR, se obţine următoarea situaţie:	HL	1114H
A	0F3H		A	0F3H
BC	0007H		BC	0004H
(1111H)	77H		(1111H)	77H
(1112H)	01H		(1112H)	01H
(1113H)	0F3H		(1113H)	0F3H
P/V	0		P/V	1
Z	0		Z	1

6. Indicatorul Z se pune pe „1" dacă A=(HL). În celelalte situaţii Z=0.

7. Executînd o operaţie de adunare a două numere, de exemplu, 17 (BCD) şi 39 (BCD), se obţine simplu, în aritmetică zecimală, rezultatul 56. Lucrînd cu reprezentări binare:

$$\begin{array}{r} 0001\ \ 0111 \\ +0011\ \ 1001 \\ \hline 0101\ \ 0000 = 50H \end{array}$$

rezultatul este incorect. Instrucţiunea DAA ajustează acest rezultat pentru a se obţine reprezentarea corectă BCD a sumei:

$$\begin{array}{r} 0101\ \ 0000 \\ +0000\ \ 0110 \\ \hline 0101\ \ 0110 = 56 \end{array}$$

8. Dacă acumulatorul conţine 0101 1101, după execuţia instrucţiunii CPL va conţine 1010 0010.

9. Conţinutul acumulatorului 0101 1101 se modifică după execuţia instrucţiunii NEG, devenind 1010 0011.

10. Valorile lui *r* sînt aceleaşi ca în tabelul 1.9; dacă *r*=110, se modifică numai indicatorii de condiţii.

11. Dacă rezultatul scăderii B-1 este „0" ,indicatorul Z se poziţionează pe „1"; în celelalte situaţii Z=0.

1.2. CIRCUITUL PENTRU COMANDA INTRĂRILOR/IEŞIRILOR PARALELE PIO-Z80

Circuitul pentru comanda intrărilor/ieşirilor paralele, PIO-Z80, este destinat conectării echipamentelor periferice cu interfeţe paralele la unităţi centrale realizate cu microprocesoare Z80.

PIO-Z80 cuplează perifericele prin intermediul a două *port*-uri de I/E de 8 biţi concepute să lucreze independent, *port*-ul A şi *port*-ul B. Fiecare *port* are asociate două semnale, READY şi $\overline{\text{STROBE}}$, cu ajutorul cărora poate comanda transferul datelor. Ieşirea READY indică perifericului faptul că *port*-ul este pregătit, gata, pentru un transfer de date. Intrarea $\overline{\text{STROBE}}$ de la periferic semnalează cînd a apărut un transfer.

Circuitul PIO-Z80 poate fi programat să lucreze în unul din următoarele patru moduri:

Modul 0, ieșire-octet
Modul 1, intrare-octet
Modul 2, intrare/ieșire-octet
Modul 3, intrare/ieșire pe bit.

Ambele *port*-uri de I/E se pot programa să lucreze în *modul 0*. Cele două *port*-uri sînt prevăzute cu registre de ieșire adresabile direct de către UC, datele putînd fi transmise către oricare din ele în orice moment. După ce datele au fost scrise într-un *port*, se va activa ieșirea READY corespunzătoare, indicîndu-se în acest fel perifericului asociat că informația e disponibilă și se poate efectua transferul. După transferarea datelor, dispozitivul extern va răspunde activînd intrarea $\overline{\text{STROBE}}$, care, la rîndul ei, va putea conduce la generarea unei întreruperi, în cazul în care aceasta a fost în prealabil validată.

În *modul 1* cele două *port*-uri ale circuitului PIO-Z80 pot fi utilizate pentru a transfera informația de la periferic spre UC. Fiecare din *port*-uri are un registru de intrare adresabil direct de unitatea centrală. Cînd urmează să se citească un octet din *port*, PIO poziționează întîi semnalul READY. Perifericul detectează acest lucru și ca răspuns, plasează datele pe liniile de intrare/ieșire, eșantionîndu-le cu ajutorul semnalului $\overline{\text{STROBE}}$. Semnalul $\overline{\text{STROBE}}$ e utilizat de PIO pentru a memora datele în registrul de intrare și a declanșa o cerere de întrerupere, dacă aceasta a fost în prealabil validată.

În *modul 2*, bidirecțional pe octet, se folosește pentru transfer un singur *port*, A, împreună cu semnalele de comandă ale ambelor *port*-uri. Celălalt *port*, B, trebuie programat în modul 3 și mascat. O operație de ieșire este similară cu un transfer în modul 0, cu observația că datele se pot genera numai cînd linia $\overline{\text{STROBE}}$ a *port*-ului A e pe „0". În intrare funcționarea este asemănătoare cu lucrul în modul 1 cu mențiunea că pentru dialog se întrebuințează semnalele de comandă și întreruperea asociate *port*-ului B.

Modul 3, utilizabil de ambele *port*-uri, permite definirea individuală a biților ca intrări sau ieșiri. Nu se folosesc semnalele de comandă, întreruperea generîndu-se la activarea unei singure intrări sau a tuturor. Prin programare se specifică nivelul activ al fiecărei intrări, „0" sau „1", și condiția logică ce va trebui să fie îndeplinită pentru a se genera întreruperea: *SAU* — o singură intrare activă, *ȘI* — toate intrările active. De exemplu, dacă unul din *port*-ur este programat să aibă intrări active pe „0" și condiția logică este *ȘI* atunci circuitul PIO-Z80 va genera o cerere de întrerupere cînd toate intrările acelui *port* trec pe „0"

Precizăm că numerotarea modurilor are și o semnificație mnemotehnică: 0=*Out* (ieșire), 1=*In*(intrare), 2=Bidirecțional.

1.2.1. STRUCTURA CIRCUITULUI PIO-Z80

În figura 1.19 este dată schema bloc a unui circuit PIO-Z80. După cum se observă, el este structurat în jurul unei magistrale interne la care sînt conectate logica de comandă internă, partea de interfață cu UC-Z80, cele două *port*-uri de I/E și logica de comandă a întreruperii.

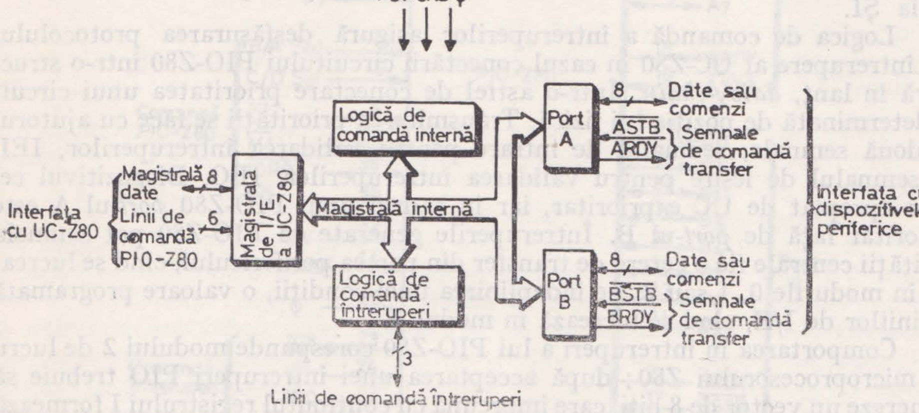


Fig. 1.19. Schema bloc a circuitului PIO-Z80

Cele două *port*-uri de I/E, identice, au organizarea din figura 1.20, conținînd fiecare registre separate de intrare și ieșire, patru registre de comandă și partea de logică pentru controlul transferului. Toate transferurile între UC și periferice utilizează registrele de intrare sau ieșire, schimbul efectiv de informație cu perifericul desfășurîndu-se sub comanda semnalelor READY și $\overline{\text{STROBE}}$. Modul de lucru al *port*-ului este memorat într-un registru de doi biți. Celelalte trei registre de comandă sînt utilizate numai în modul 3.

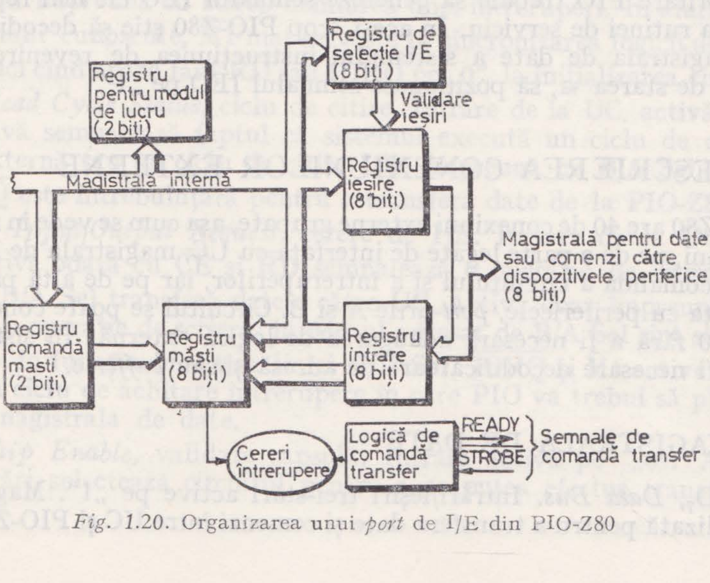


Fig. 1.20. Organizarea unui *port* de I/E din PIO-Z80

Registrul de selecţie I/E specifică biţii de ieşire, validîndu-i; biţii rămaşi sînt consideraţi intrări. Registrele de măşti şi de comandă-măşti se folosesc pentru fixarea condiţiilor de întrerupere. Registrul de măşti memorează biţii activi şi pe cei inactivi sau mascaţi. Cu ajutorul registrului de comandă-măşti se precizează întîi dacă starea activă a biţilor este „1" sau „0", apoi dacă întreruperea se va genera cînd oricare din biţii nemascaţi este activ, condiţia SAU, ori numai cînd toţi biţii de intrare nemascaţi sînt activi, condiţia ŞI.

Logica de comandă a întreruperilor asigură desfăşurarea protocolului de întrerupere al UC-Z80 în cazul conectării circuitului PIO-Z80 într-o structură în lanţ, *daisy-chain*. Într-o astfel de conectare prioritatea unui circuit e determinată de poziţia lui fizică. Transmiterea priorităţii se face cu ajutorul a două semnale, semnalul de intrare pentru validarea întreruperilor, IEI, şi semnalul de ieşire pentru validarea întreruperilor, IEO. Dispozitivul cel mai apropiat de UC e prioritar, iar în cadrul unui PIO-Z80 *port*-ul A este prioritar faţă de *port*-ul B. Întreruperile generate cu PIO-Z80 pot semnala unităţii centrale fie o cerere de transfer din partea perifericului, cînd se lucrează în modurile 0, 1 sau 2, fie îndeplinirea unei condiţii, o valoare programată a liniilor de I/E, cînd se lucrează în modul 3.

Comportarea în întreruperi a lui PIO-Z80 corespunde modului 2 de lucru al microprocesorului Z80; după acceptarea unei întreruperi PIO trebuie să genereze un vector de 8 biţi, care împreună cu conţinutul registrului I formează adresa unei locaţii de unde se va citi adresa subrutinei de serviciu. Fiecare *port* poate fi programat să aibă un vector de întrerupere propriu. Cel mai puţin semnificativ bit al vectorului este întotdeauna pus pe zero în PIO-Z80, avînd în vedere că adresa pe 16 biţi a subrutinei de serviciu trebuie să se găsească în două locaţii succesive de memorie începînd cu o adresă pară.

Spre deosebire de alte circuite periferice din familia Z80, PIO nu validează întreruperile imediat după programare ci mai tîrziu, după ce a fost sesizată o trecere a lui $\overline{M1}$ pe „0", corespunzătoare, de exemplu, unui ciclu de extragere. Pentru a permite tratarea întreruperilor de la dispozitivele mai puţin prioritare PIO trebuie să genereze semnalul IEO cît mai repede după terminarea rutinei de serviciu. În acest scop PIO-Z80 ştie să decodifice direct de pe magistrala de date a sistemului instrucţiunea de revenire RETI şi în funcţie de starea sa, să poziţioneze semnalul IEO pe „1".

1.2.2. DESCRIEREA CONEXIUNILOR EXTERNE

PIO-Z80 are 40 de conexiuni externe grupate, aşa cum se vede în figura 1.21 pe funcţiuni, pe de o parte legate de interfaţa cu UC, magistrala de date, semnalele de comandă a circuitului şi a întreruperilor, iar pe de altă parte legate de interfaţa cu perifericele, *port*-urile A şi B. Circuitul se poate conecta direct cu UC-Z80 fără a fi necesară adăugarea de logică externă. În sisteme mari ar putea fi necesare decodificatoare de adresă şi/sau *buffer*-e.

1.2.2.1. MAGISTRALA DE DATE

$D_0 \div D_7$, *Data Bus*. Intrări/ieşiri trei-stări active pe „1". Magistrala de date e utilizată pentru a transfera date şi comenzi între UC şi PIO-Z80.

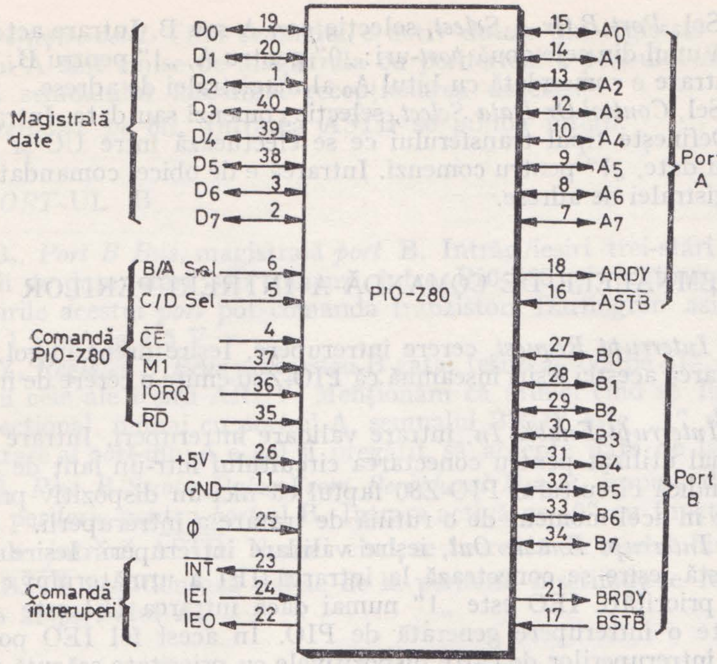


Fig. 1.21. Conexiunile externe ale circuitului PIO-Z80

1.2.2.2. SEMNALELE DE COMANDĂ A CIRCUITULUI

$\overline{M1}$, *Machine Cycle One,* primul ciclu-mașină. Intrare de la UC activă pe „0". Este folosită ca impuls de sincronizare pentru a comanda diferite operații interne din PIO. $\overline{M1}$ activ împreună cu $\overline{RD}$ semnalează un ciclu de extragere, iar împreună cu $\overline{IORQ}$ achitarea unei întreruperi. În afara acestor două funcțiuni cunoscute, $\overline{M1}$ mai servește la sincronizarea logicii de întreruperi și, atunci cînd apare fără $\overline{RD}$ sau $\overline{IORQ}$ pe „0" la inițializarea circuitului.

$\overline{RD}$, *Read Cycle Status,* ciclu de citire. Intrare de la UC, activă pe „0". Cînd e activă semnalează faptul că sistemul execută un ciclu de citire din memoria externă sau un ciclu de citire I/E. Împreună cu B/A Sel, C/D Sel, $\overline{CE}$ și $\overline{IORQ}$ este întrebuințată pentru a transfera date de la PIO-Z80 la UC.

$\overline{IORQ}$, *Input/Output Request,* cerere de I/E. Intrare de la UC, activă pe „0". Activă odată cu $\overline{CE}$ și $\overline{RD}$ semnalează o operație de citire: *port*-ul adresat de B/A Sel transferă datele către UC. Activă doar împreună cu $\overline{CE}$, semnalează o operație de scriere: în *port*-ul adresat de B/A Sel sînt scrise date sau comenzi în funcție de starea liniei C/D Sel. $\overline{IORQ}$ și $\overline{M1}$ active simultan specifică un ciclu de achitare întrerupere în care PIO va trebui să plaseze un vector pe magistrala de date.

$\overline{CE}$, *Chip Enable,* validare capsulă. Intrare activă pe „0". Activarea acestei intrări selectează circuitul pentru a se putea efectua transferuri de date cu UC.

B/A Sel, *Port B Or A Select*, selecţie *port* A sau B. Intrare activă pe „1" Selectează unul din cele două *port*-uri: „0" pentru A, „1" pentru B. De obicei această intrare e comandată cu bitul A_0 al magistralei de adrese.

C/D Sel, *Control Or Data Select*, selecţie comenzi sau date. Intrare activă, pe „1". Defineşte tipul transferului ce se efectuează între UC şi PIO-Z80:. „0" pentru date, „1" pentru comenzi. Intrarea e de obicei comandată cu bitul A_1 al magistralei de adrese.

1.2.2.3. SEMNALELE DE COMANDĂ A ÎNTRERUPERILOR

$\overline{\text{INT}}$, *Interrupt Request*, cerere întrerupere. Ieşire drenă-în-gol activă pe „0". Activarea acestei ieşiri înseamnă că PIO-Z80 emite o cerere de întrerupere către UC.

IEI, *Interrupt Enable In*, intrare validare întreruperi. Intrare activă pe „1". Semnal utilizat pentru conectarea circuitului într-un lanţ de priorităţi: IEI = 1 indică circuitului PIO-Z80 faptul că nici-un dispozitiv prioritar nu este servit în acel moment de o rutină de tratare a întreruperii.

IEO, *Interrupt Enable Out*, ieşire validare întreruperi. Ieşire activă pe „1". Această ieşire se conectează la intrarea IEI a următorului circuit din lanţul de priorităţi. IEO este „1" numai dacă intrarea IEI este „1" şi UC nu serveşte o întrerupere generată de PIO. În acest fel IEO poate bloca generarea întreruperilor de către dispozitivele cu prioritate scăzută pe timpul tratării unei întreruperi solicitate de un circuit prioritar.

1.2.2.4. *PORT*-UL A

$A_0 \div A_7$, *Port A Bus*, magistrală *port* A. Intrări/ieşiri trei-stări prin intermediul cărora se transferă date, stări sau comenzi între PI0-Z80 şi echipamentul periferic conectat.

ARDY, *Register A Ready*, registru A gata. Ieşire activă pe „1" a cărei semnificaţie depinde de modul de lucru programat:

Ieşire-octet. Este activată cînd registrul de ieşire al *port*-ului A a fost încărcat, magistrala cu perifericul e stabilă şi pregătită pentru transferul octetului la periferic.

Intrare-octet. ARDY e activată cînd registrul de intrare al *port*-ului A este gol şi gata să accepte date de la periferic.

Intrare/ieşire-octet. Ieşirea este activată cînd datele sînt disponibile în registrul de ieşire al *port*-ului A pentru a fi transferate către periferic. În acest mod datele nu vor fi plasate pe magistrala cu perifericul pînă cînd $\overline{\text{ASTB}}$ nu e activ.

Intrare/ieşire pe bit. ARDY e invalidat şi forţat pe „0".

$\overline{\text{ASTB}}$, *Port A Strobe Pulse From Peripheral Device*, impuls de eşantionare de la periferic pentru *port*-ul A. Intrare activă pe „0" avînd următoarele semnificaţii, în funcţie de modul de lucru:

Ieşire-octet. Frontul pozitiv al acestui semnal este generat de periferic pentru a semnala recepţionarea datelor emise de PIO.

Intrare-octet. Se emite de periferic pentru a încărca datele în registrul de intrare al *port*-ului A. Încărcarea se face cînd semnalul e activ.

Intrare/ieşire-octet. Cînd semnalul e activ datele din registrul de ieşire al *port*-ului A sînt emise pe magistrala cu perifericul a *port*-ului A. Frontul pozitiv al semnalului înseamnă recepţionarea datelor.

Intrare/ieşire pe bit. Intrarea $\overline{ASTB}$ se inhibă intern.

1.2.2.5. *PORT*-UL B

$B_0 \div B_7$, *Port B Bus,* magistrală *port* B. Intrări/ieşiri trei-stări destinate transferării de date, stări sau comenzi între PI0-Z80 şi echipamente periferice. Ieşirile acestui *port* pot comanda tranzistori Darlington asigurînd un curent de 1,5 mA la 1,5 V.

BRDY, *Register B Ready,* registru B gata. Ieşire activă pe „1" cu funcţii similare cu cele ale ieşirii ARDY. Menţionăm că atunci cînd se lucrează în mod bidirecţional, numai cu *port*-ul A, semnalul BRDY este „1" dacă registrul de intrare al *port*-ului A e gol şi pregătit să accepte date de la periferic.

$\overline{BSTB}$, *Port B Strobe Pulse From Peripheral Device,* impuls de eşantionare de la periferic pentru *port*-ul B. Intrare activă pe „0" cu funcţii similare cu cele ale intrării $\overline{ASTB}$. Numai cînd se lucrează în modul bidirecţional semnalul $\overline{BSTB}$ eşantionează datele de la periferic înscriindu-le în registrul de intrare al *port*-ului A.

1.2.3. DIAGRAME DE TIMP

1.2.3.1. CICLUL DE SCRIERE UC

Scrierea comenzilor sau datelor în PIO-Z80 se face cu ajutorul instrucţiunilor de ieşire. Diagrama de timp a unui ciclu de scriere este dată în figura 1.22. Se observă concordanţa cu ciclul de ieşire al microprocesorului Z80 (vezi figura 1.10) care introduce în mod automat o singură stare de aşteptare T_w. Pentru a se asigura funcţionarea corectă a circuitului PIO, nu se mai admite inserarea altor stări T_w. Menţionăm, de asemenea, că neexistînd o conexiune externă pe care să primească un semnal explicit de comandă a scrierii, PIO-Z80 îl generează intern. Acest semnal, $\overline{WR}$* în figura 1.22, generat intern, este activ cînd lipseşte $\overline{RD}$.

1.2.3.2. CICLUL DE CITIRE UC

Diagrama de timp, prezentată în figura 1.23, arată, ca şi în cazul scrierii, concordanţa cu ciclul de intrare al microprocesorului Z80. La fel, nu se permite introducerea de stări T_w suplimentare faţă de starea adăugată în mod automat de UC-Z80.

1.2.3.3. MODUL 0 (IEŞIRE-OCTET)

Desfăşurarea în timp a operaţiilor specifice modului 0 de lucru, ieşire-octet, este ilustrată în figurile 1.24, 1.25 şi 1.26. Iniţierea unui ciclu de ieşire se face întotdeauna prin execuţia, de către UC, a unei instrucţiuni OUT.

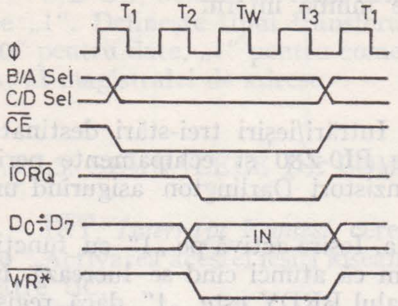


Fig. 1.22. Ciclul de scriere UC in PIO-Z80

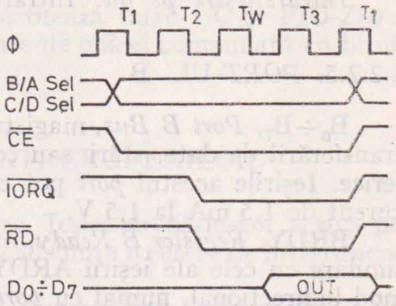


Fig. 1.23. Ciclul de citire UC din PIO-Z80

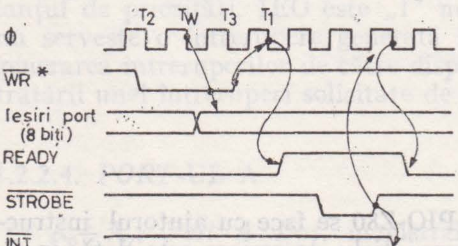


Fig. 1.24. Ieșire-octet utilizînd ambele semnale READY și STROBE

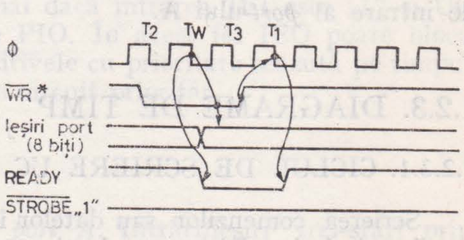


Fig. 1.25. Ieșire-octet cu intrarea STROBE conectată la „1"

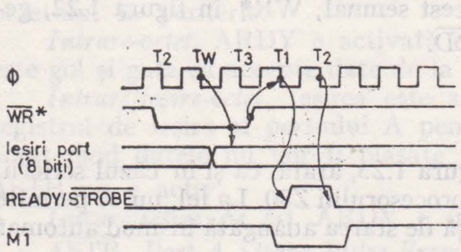


Fig. 1.26. Ieșire-octet cu liniile READY și STROBE legate împreună

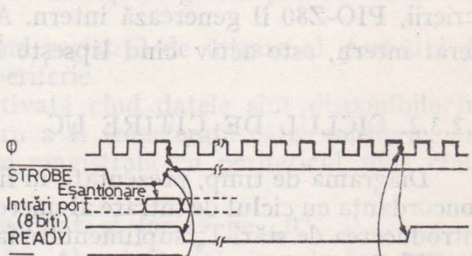


Fig. 1.27. Intrare-octet utilizînd ambele semnale READY și STROBE

Impulsul $\overline{WR}$*, generat intern pe timpul unei astfel de instrucţiuni, va condiţiona înscrierea datelor trimise pe magistrala de date a sistemului, $D_0 \div D_7$, în registrul de ieşire al *port*-ului adresat. La sfîrşitul instrucţiunii de ieşire, după dezactivarea lui $\overline{WR}$*, PIO va poziţiona pe „1", cu frontul negativ al ceasului Φ, ieşirea READY corespunzătoare *port*-ului în care s-a făcut scrierea. În acest fel se indică perifericului că poate prelua un octet, frontul pozitiv al semnalului READY putînd fi folosit de către dispozitivul de I/E pentru memorarea acestui octet. READY va rămîne activ, „1", pînă la recepţionarea unui front pozitiv pe linia $\overline{STROBE}$ cu ajutorul căruia se va genera totodată, o întrerupere, dacă *port*-ul respectiv a fost programat şi validat să lucreze în întreruperi şi dacă este prioritar.

În cazul în care se lucrează fără semnalul $\overline{STROBE}$, cu intrarea corespunzătoare legată la „1", ieşirea READY va fi forţată pe „0" după o perioadă şi jumătate a lui Φ de la activarea comenzii $\overline{IORQ}$. READY va fi repus pe „1" cu primul front negativ al ceasului după dezactivarea lui $\overline{IORQ}$ (figura 1.25). Această acţionare garantează că READY este întotdeauna pe „0" la schimbarea datelor în *port* şi că trecerea lui pe „1" se face ori de cîte ori se va executa o instrucţiune OUT.

Dacă cele două linii de comandă a transferului, READY şi $\overline{STROBE}$, sînt legate împreună, PIO-Z80 va genera pe aceste linii un impuls cu durata o perioadă a ceasului Φ ca în figura 1.26. Frontul pozitiv al lui READY/$\overline{STROBE}$ nu va mai poziţiona pe „0" ieşirea $\overline{INT}$, datorită logicii interne din PIO, care nu permite ca durata impulsului pozitiv pe $\overline{STROBE}$ să fie mai mică decît durata lui $\overline{M1}$.

1.2.3.4. MODUL 1 (INTRARE-OCTET)

Un ciclu de intrare poate fi iniţiat de către periferic prin punerea pe „0" a liniei $\overline{STROBE}$ corespunzătoare *port*-ului programat să lucreze în modul 1, ca în figura 1.27. Pe timpul cît acest semnal este „0" intrările în *port* sînt eşantionate, transferîndu-se în registrul de intrare al *port*-ului. Cu frontul pozitiv al lui $\overline{STROBE}$ se poate genera o întrerupere, ca şi în modul 0, dacă *port*-ul respectiv a fost programat şi validat să lucreze în întreruperi şi dacă este prioritar. După trecerea pe „1" a lui $\overline{STROBE}$, cu următorul front negativ al lui Φ, se va inactiva ieşirea READY semnalînd perifericului că registrul de intrare este plin şi că trebuie inhibată încărcarea unui nou octet, pînă la preluarea de către UC a octetului păstrat în registrul de intrare al *port*-ului. Preluarea se face cu ajutorul unei instrucţiuni IN de obicei în timpul rutinei de tratare a întreruperii. Activarea semnalului READY se va face la sfîrşitul acestei instrucţiuni, cu primul front negativ, al ceasului Φ după trecerea pe „1" a lui $\overline{RD}$, indicîndu-se astfel dispozitivului de I/E că poate transmite un nou octet către PIO.

Ţinînd cont că după iniţializare ieşirea READY rămîne pe „0", pentru a lansa un transfer în modul 1, este necesară, în unele sisteme, execuţia unei instrucţiuni IN false pentru a se face READY=1. Vom detalia această observaţie în § 1.2.4.

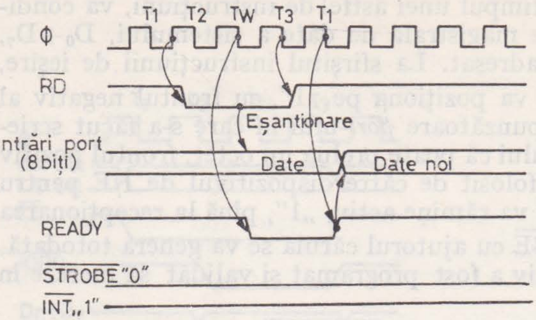


Fig. 1.28. Intrare-octet cu linia STROBE conectată la „0"

Dacă nu se folosește, intrarea STROBE trebuie conectată la „0". În această situație semnalul READY va fi inactivat o perioadă și jumătate a ceasului Φ după trecerea pe „0" a comenzii IORQ în timpul unei instrucțiuni IN de citire a *port*-ului (figura 1.28). Astfel se previne schimbarea datelor în registrul de intrare pe timpul eșantionării acestuia de către UC. READY va fi reactivat la fel ca mai sus cu primul front negativ al lui Φ după trecerea pe „1" a lui RD.

1.2.3.5. MODUL 2 (INTRARE/IEȘIRE-OCTET)

În figura 1.29 este dată desfășurarea în timp a transferului de octeți cînd PIO-Z80 este programat să lucreze în modul 2. Acest mod poate fi programat numai pentru *port*-ul A, utilizîndu-se toate semnalele de comandă-transfer ale circuitului PIO.

După cum se poate vedea în figura 1.29, funcționarea este aproape identică cu funcționarea corespunzătoare modurilor 0 și 1, descrisă anterior (vezi figurile 1.24, 1.27): pentru comanda ieșirii se folosesc liniile ARDY și ASTB, iar pentru intrare BRDY și BSTB. Diferența constă numai în faptul că, atunci cînd se lucrează în modul 2/ieșire-octet, datele sînt plasate pe magistrala către periferic numai pe durata cît ASTB este activ, dispozitivul de I/E putînd intrebuința frontul pozitiv al acestui semnal pentru memorarea octetului.

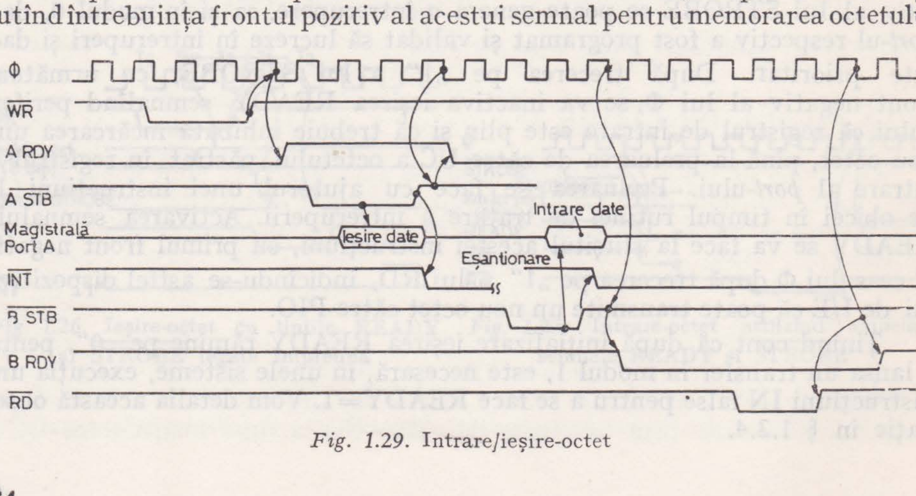


Fig. 1.29. Intrare/ieșire-octet

În timp ce $\overline{ASTB}$ este activ, adică atunci cînd PIO efectuează o ieșire-octet, dispozitivul periferic nu trebuie să plaseze date pe magistrala de intrare/ieșire a *port*-ului A. În scopul rezolvării acestui conflict potențial perifericul poate utiliza semnalul $\overline{BSTB}$, decalat în timp față de $\overline{ASTB}$, pentru a valida emisia datelor pe magistrală. Această condiționare este permisă ținînd seama că PIO-Z80, atunci cînd lucrează în modul 2, memorează datele în registrul de intrare cînd $\overline{BSTB}$=0, pe nivel. De asemenea, validarea pe magistrală a datelor cu $\overline{BSTB}$ este permisă și pentru că PIO-Z80 nu impune nici-un timp de menținere a datelor după dezactivarea lui $\overline{BSTB}$. Dacă $\overline{ASTB}$ este activ pe timpul cît UC execută o operație de citire a *port*-ului A, ca răspuns la o întrerupere generată cu $\overline{BSTB}$, UC va citi registrul de ieșire în locul registrului de intrare. Deși datele eșantionate cu $\overline{BSTB}$ sînt, în această situație, corect memorate în registrul de intrare al *port*-ului, UC nu poate citi registrul. Pentru a elimina această situație, activarea lui $\overline{ASTB}$ poate fi condiționată și de BRDY=1, avînd în vedere că, atunci cînd BRDY=0, PIO-Z80 poate aștepta rezolvarea de către UC a unei întreruperi generate cu $\overline{BSTB}$.

1.2.3.6. MODUL 3 (INTRARE/IEȘIRE PE BIT)

Cînd PIO-Z80 funcționează în modul 3 semnalele de comandă a transferului nu sînt folosite, unitatea centrală a sistemului putînd să execute în orice moment operații de scriere sau citire în/din *port*-uri. La scriere datele sînt memorate în registrul de ieșire cu aceeași desfășurare în timp ca în modul 0. Așa cum am spus la descrierea conexiunilor externe, ARDY este întotdeauna forțat pe „0" cînd *port*-ul A lucrează în modul 3. Același lucru se întîmplă și cu BRDY, excepție făcînd situația cînd *port*-ul A este în modul 2, ieșirea BRDY rămînînd în acest caz neafectată.

În figura 1.30 este dată desfășurarea în timp a unei operații de citire a unui *port* programat să lucreze în modul 3. Octetul citit de către UC va fi compus din biți aparținînd registrului de ieșire, pentru liniile desemnate ca ieșiri, și din biți aparținînd registrului de intrare, pentru liniile specificate ca intrări. Registrul de intrare va conține datele prezente pe magistrala cu

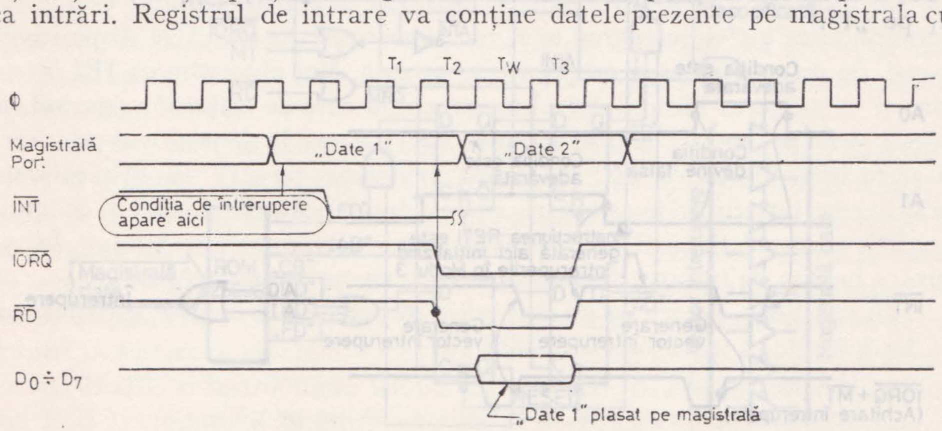


Fig. 1.30. Citire dintr-un *port* programat în modul 3

perifericul înainte de frontul negativ al semnalului $\overline{RD}$: în figura 1.30 UC va citi „Date 1", condiția care a generat întreruperea, și nu „Date 2".

Cînd se lucrează în modul 3 PIO supraveghează în permanență liniile nemascate ale *port*-ului pentru a surprinde apariția unei configurații de biți stabilite să genereze o întrerupere, evident numai dacă întreruperile corespunzătoare *port*-ului respectiv sînt validate. Întreruperea se generează atunci cînd condiția logică se schimbă din falsă în adevărată. Pentru ca același *port* să genereze o nouă întrerupere, condiția care a generat întreruperea trebuie să devină falsă și apoi, din nou, adevărată. De exemplu, să presupunem condiția logică *SAU*. Întreruperea va fi generată dacă o linie nemascată devine activă. Dacă, în continuare, o altă linie nemascată devine și ea activă, PIO-Z80 *nu va mai genera o nouă întrerupere*. Mai mult, dacă două linii nemascate devin active simultan, PIO va genera o singură întrerupere. Înainte ca PIO să genereze o altă întrerupere trebuie, deci, ca toate liniile nemascate să devină inactive și apoi cel puțin una să fie activată. Conexiunile externe ale *port*-ului definite ca ieșiri pot contribui la condiția logică ce generează întreruperea dacă pozițiile lor nu sînt mascate.

În cazul în care condiția logică devine adevărată puțin înainte de $\overline{M1}$ sau pe durata lui $\overline{M1}$, întreruperea va fi generată după frontul pozitiv al acestui semnal, dacă această condiție rămîne adevărată și după front.

În figura 1.31 se dă un exemplu de generare a întreruperii de către *port*-ul A programat să lucreze în modul 3. Condiția logică selectată este *SAU* iar starea activă „1". Toți biții sînt mascați în afară de A_0 și A_1, ceea ce echivalează deci cu o poartă *SAU* în logică pozitivă cu două intrări. După poziționarea lui A_0 pe „1", condiția logică devine adevărată, PIO generînd o întrerupere. UC răspunde la această întrerupere cu un ciclu de achitare-întrerupere, $\overline{IORQ}+\overline{M1}=0$, în care PIO va transmite vectorul de întrerupere și UC va apela subrutina de serviciu corespunzătoare. A_0 poate trece acum pe „0" fie singur, fie ca rezultat al unei acțiuni întreprinse în subrutina de serviciu. Trecerea lui A_0 pe „0" înseamnă, în situația din figura 1.31 cînd și A_1 e „0", o anulare a condiției logice care generează întreruperea. Acum, după execuția instrucțiunii de revenire RETI care inițializează schema de întreruperi din PIO, o nouă întrerupere poate fi activată, de exemplu, prin poziționarea lui A_1 pe „1".

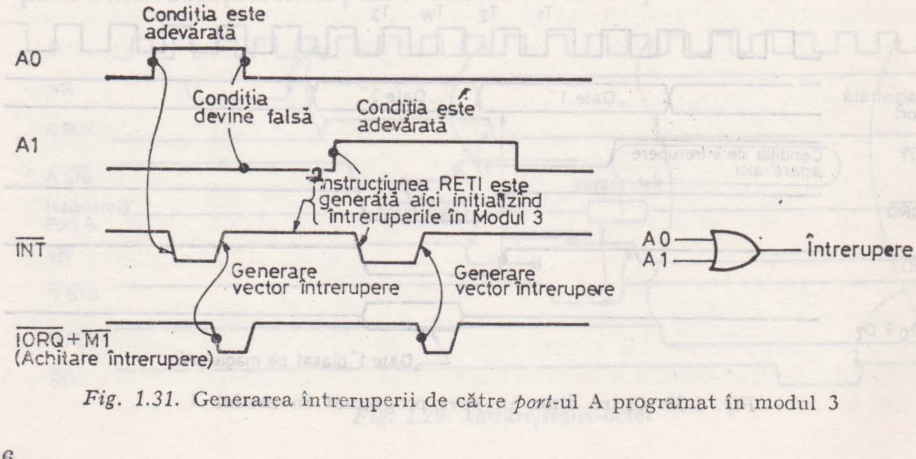


Fig. 1.31. Generarea întreruperii de către *port*-ul A programat în modul 3

Din acest exemplu reținem deci, în primul rînd, că pentru ca A_1 să genereze o întrerupere, semnalul nu trebuie să devină „1" decît după ce A_0 a trecut pe „0". De asemenea, pentru ca A_1 să genereze o întrerupere, va trebui să se poziționeze pe „1" după execuția instrucțiunii RETI din subrutina de tratare a întreruperii generate de A_0. În concluzie, pentru a se genera o nouă întrerupere, condiția logică trebuie să devină falsă după achitarea întreruperii precedente și să redevină adevărată după ce RETI inițializează schema de întreruperi din PIO-Z80.

1.2.3.7. INTRERUPERI

Pentru o mai bună înțelegere a comportării în intreruperi a circuitului PIO-Z80 dăm în figura 1.32 schema generală a blocului de comandă a întreruperilor implementat, cu mici adaptări, în toate circuitele periferice ale familiei Z80. Această schemă poate fi utilizată și în cazul interfațării microprocesorului Z80 cu circuite periferice care nu au prevăzut mecanismul de lucru în intreruperi specific familiei Z80 (modul 2).

Funcțiile realizate de blocul de comandă a întreruperilor sînt: activarea semnalului de întrerupere $\overline{INT}$ către unitatea centrală, comanda ieșirii IEO, plasarea pe magistrala de date a vectorului în ciclul de achitare a între-

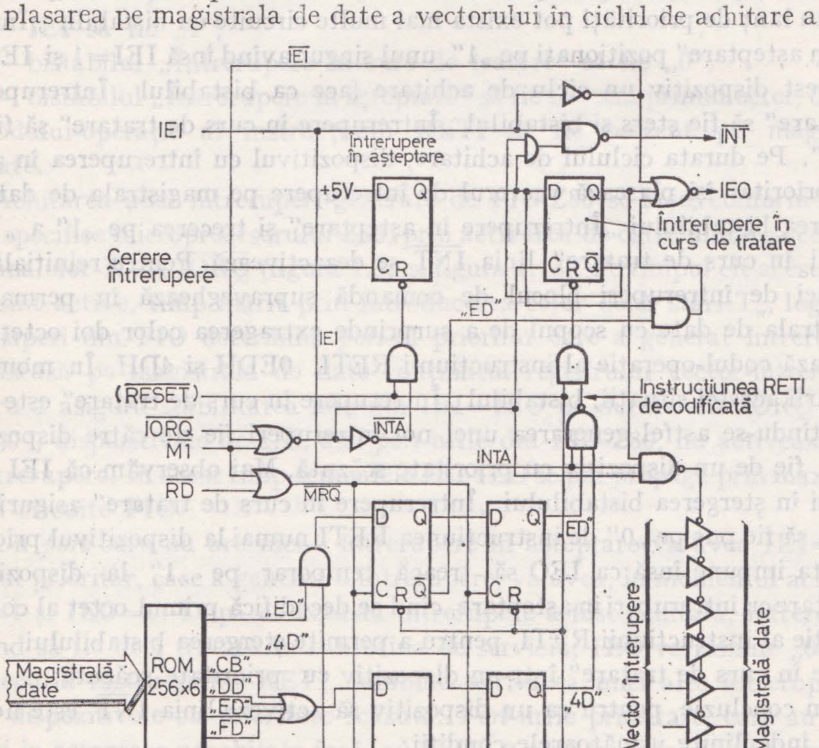


Fig. 1.32. Schema blocului de comandă a intreruperilor

ruperii, decodificarea instrucțiunii RETI în vederea reinițializării schemei. Corespunzător acestor funcții, blocul este alcătuit din bistabilii „Întrerupere în așteptare“ și „Întrerupere în curs de tratare“, alți patru bistabili și o memorie ROM pentru decodificarea instrucțiunii RETI, circuite combinaționale de comandă.

La inițializarea circuitului bistabilii se șterg, ceea ce conduce la dezactivarea liniei $\overline{INT}$ și la IEO=IEI. Pentru generarea unei întreruperi spre UC este necesar, mai întii, să se activeze semnalul intern „Cerere-întrerupere“ specific fiecărui circuit periferic din familia Z80. Activarea acestui semnal conduce, după cum se vede în figura 1.32, la poziționarea pe „1“ a bistabilului „Întrerupere în așteptare“. Se observă, de asemenea, că poziționarea pe „1“ a bistabilului se inhibă la dispozitive prioritare, cu IEI=1, pe durata unui ciclu de achitare. În acest fel se asigură înghețarea lanțului de priorități spre prioritatea înaltă pe timpul unui ciclu de achitare a întreruperii. În continuare, pentru ca dispozitivul să activeze semnalul $\overline{INT}$, trebuie îndeplinite simultan condițiile: IEI=1, bistabilul „Întrerupere în așteptare“ să fie pe „1“, bistabilul „Întrerupere în curs de tratare“ pe „0“. Totodată poziționarea pe „1“ a bistabilului „Întrerupere în așteptare“ duce la punerea pe zero a ieșirii IEO. UC răspunde la întrerupere cu un ciclu de achitare în care $\overline{IORQ}=\overline{M1}=0$. Într-un lanț de priorități pot exista mai multe circuite cu bistabilii „Întrerupere în așteptare“ poziționați pe „1“, unul singur avînd însă IEI=1 și IEO=0. La acest dispozitiv un ciclu de achitare face ca bistabilul „Întrerupere în așteptare“ să fie șters și bistabilul „Întrerupere în curs de tratare“ să fie pus pe „1“. Pe durata ciclului de achitare, dispozitivul cu întreruperea în așteptare prioritar își plasează vectorul de întrerupere pe magistrala de date. La ștergerea bistabilului „Întrerupere în așteptare“ și trecerea pe „1“ a „Întreruperii în curs de tratare“ linia $\overline{INT}$ se dezactivează. Pentru reinițializarea schemei de întreruperi blocul de comandă supraveghează în permanență magistrala de date cu scopul de a surprinde extragerea celor doi octeți care formează codul-operație al instrucțiunii RETI, 0EDH și 4DH. În momentul decelării acestei situații, bistabilul „Întrerupere în curs de tratare“ este șters, permițîndu-se astfel generarea unei noi întreruperi fie de către dispozitivul însuși, fie de un dispozitiv cu prioritate scăzută. Mai observăm că IEI intervine și în ștergerea bistabilului „Întrerupere în curs de tratare“ asigurînd ca acesta să fie pus pe „0“ de instrucțiunea RETI numai la dispozitivul prioritar. Aceasta impune însă ca IEO să treacă temporar pe „1“ la dispozitivele prioritare cu întreruperi în așteptare, cînd se decodifică primul octet al codului-operație al instrucțiunii RETI, pentru a permite ștergerea bistabilului „Întrerupere în curs de tratare“ într-un dispozitiv cu prioritate scăzută.

În concluzie, pentru ca un dispozitiv să activeze linia $\overline{INT}$, este necesar să fie îndeplinite următoarele condiții:

— IEI să fie „1“,
— bistabilul „Întrerupere în așteptare“ să fie pe „1“,
— bistabilul „Întrerupere în curs de tratare“ să fie pe „0“.

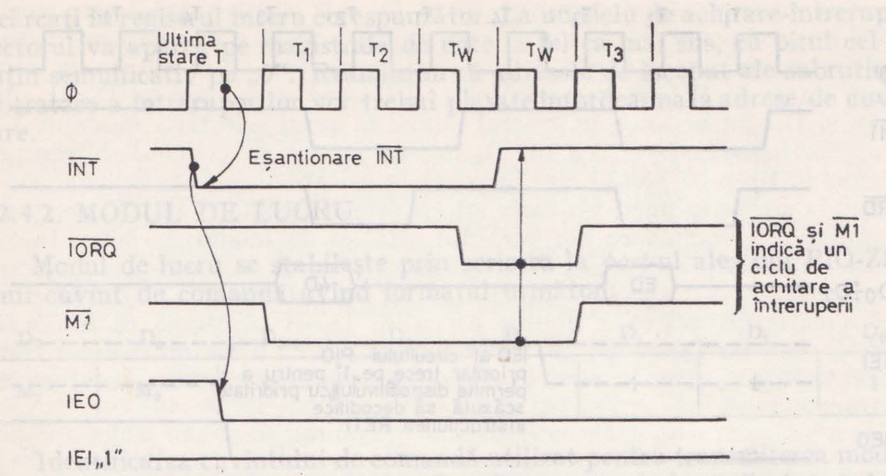


Fig. 1.33. Achitarea unei întreruperi generate de PIO-Z80

Pentru ca ieşirea IEO să fie pe „1" trebuie îndeplinite condiţiile:

— IEI să fie „1",

— bistabilul „Întrerupere în curs de tratare" să fie „0",

— bistabilul „Întrerupere în aşteptare" să fie „0" sau primul octet, 0EDH, al codului-operaţie al instrucţiunii RETI să fie prezent pe magistrala de date.

Achitarea unei întreruperi generate de PIO-Z80 se face, conform procedurii specifice microprocesorului Z80, prin activarea de către unitatea centrală a semnalelor $\overline{M1}$ şi $\overline{IORQ}$ (figura 1.33 şi figura 1.13). Pe timpul cît aceste semnale sînt active, timp mărit prin introducerea celor două stări T_w, logica de întreruperi din PIO determină *port*-ul prioritar care a generat întreruperea şi plasează pe magistrala de date conţinutul registrului vector-întrerupere. Pentru a asigura stabilitatea lanţului IEI=IEO pe durata cît $\overline{IORQ}$ şi $\overline{M1}$ sînt „0", dispozitivele de I/E, aici *port*-urile din PIO-Z80, nu activează linia de întrerupere. În acest timp semnalele IEI-IEO se pot propaga prin maximum patru circuite PIO.

Un *port* care nu are nici-o întrerupere în aşteptare va avea IEI=IEO. *Port*-ul prioritar, care a generat o întrerupere, va avea, în momentul achitării, IEI=1 şi IEO=0. După ce această întrerupere a fost achitată, întreruperea urmînd să fie deci tratată în subrutina de serviciu, IEO va rămîne „0" pînă la execuţia instrucţiunii RETI, inhibînd activarea unei alte întreruperi de către dispozitivele cu prioritate scăzută. *Port*-urile prioritare, care au întreruperi în aşteptare neachitate încă, au IEO=0 pînă la decodificarea primului octet, 0EDH, al instrucţiunii RETI, (figura 1.34). În această situaţie, semnalul IEO al circuitului prioritar va trece temporar pe „1", permiţînd dispozitivului cu prioritate scăzută avînd întreruperea în curs de tratare, ca, deco-

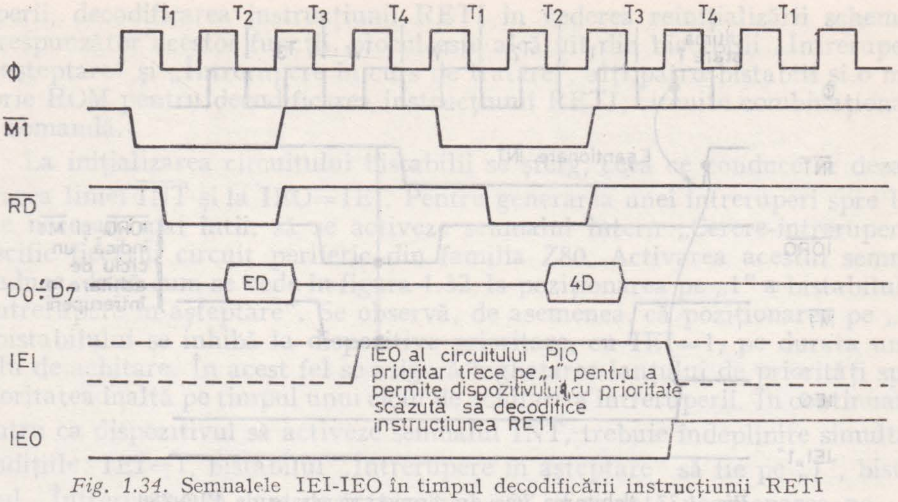


Fig. 1.34. Semnalele IEI-IEO în timpul decodificării instrucțiunii RETI

dificînd al doilea octet al codului-operație al instrucțiunii RETI, să-și șteargă bistabilul „Întrerupere în curs de tratare" și să treacă propriul semnal IEO pe „1".

1.2.4. PROGRAMAREA CIRCUITULUI PIO-Z80

Prin programarea lui PIO-Z80 se specifică următoarele trei grupe importante de parametri ce caracterizează funcționarea ulterioară a circuitului:
— vectorul de întrerupere,
— modul de lucru,
— cuvîntul de comandă-întreruperi.

12.4.1. VECTORUL DE ÎNTRERUPERE

PIO-Z80 a fost proiectat să lucreze conform protocolului corespunzător modului 2 de lucru în întreruperi al microprocesorului Z80 (vezi § 1.1). Acest protocol impune ca dispozitivul care întrerupe să genereze un vector de întrerupere, utilizat apoi de unitatea centrală ca octet mai puțin semnificativ al adresei de unde se va citi adresa subrutinei de serviciu. Vectorul de întrerupere este plasat pe magistrala de date, în timpul unui ciclu de achitare întrerupere, așa cum am arătat în § 1.2.3. Vectorul de întrerupere dorit se încarcă în PIO prin scrierea în *port*-ul ales a unui cuvînt de comandă ce are următorul format:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
V_7	V_6	V_5	V_4	V_3	V_2	V_1	0

$D_0 = 0$, într-un cuvînt de comandă, semnifică circuitului PIO-Z80 că octetul respectiv este un vector de întrerupere și deci că biții $V_7 \div V_1$ trebuie

încărcaţi în registrul intern corespunzător. La un ciclu de achitare-întrerupere, vectorul va apărea pe magistrala de date la fel ca mai sus, cu bitul cel mai puţin semnificativ pe „0". Reamintim că adresele de început ale subrutinelor de tratare a întreruperilor vor trebui plasate întotdeauna la adrese de cuvînt, pare.

1.2.4.2. MODUL DE LUCRU

Modul de lucru se stabileşte prin scrierea la *port*-ul ales din PIO-Z80 a unui cuvînt de comandă avînd formatul următor:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
M_1	M_0	X	X	1	1	1	1

Identificarea cuvîntului de comandă utilizat pentru transmiterea modului de lucru se face cu ajutorul biţilor $D_3 \div D_0$, care trebuie să fie 1 1 1 1. Cei mai semnificativi biţi, M_1 şi M_0, stabilesc modul, după cum urmează:

M_1	M_0	Modul
0	0	0 (ieşire-octet)
0	1	1 (intrare-octet)
1	0	2 (intrare/ieşire-octet, bidirecţional)
1	1	3 (intrare/ieşire pe bit)

Biţii D_5 şi D_4 sînt ignoraţi, putînd avea deci, orice valoare.

Dacă se selectează modul 3, următorul cuvînt de comandă transmis către PIO trebuie să definească liniile *port*-ului ca intrări sau ieşiri. Formatul acestui cuvînt de comandă este:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
I/E_7	I/E_6	I/E_5	I/E_4	I/E_3	I/E_2	I/E_1	I/E_0

$I/E_i = 1$ defineşte linia *i* a *port*-ului ca intrare iar $I/E_i = 0$ ca ieşire.

1.2.4.3. CUVÎNTUL DE COMANDĂ-ÎNTRERUPERI

Comanda întreruperilor se face pentru fiecare din *port*-uri cu un cuvînt specific al cărui format este:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Validare întreruperi	ŞI/SAU	„1"/„0"	Urmează măştile	0	1	1	1

Cuvîntul de comandă-întreruperi se recunoaşte după cei mai puţin semnificativi patru biţi care trebuie să fie 0 1 1 1. Dacă bitul $D_7 = 1$, bistabilul intern de validare a întreruperilor pentru *port*-ul respectiv este pus pe „1"

și *port*-ul poate genera întreruperi. Dacă $D_7 = 0$ acest bistabil este șters și *port*-ul nu mai poate activa linia de întreruperi. Dacă o întrerupere apare în timp ce se transmite cuvîntul de comandă-întreruperi cu $D_7=0$ ea va fi memorată intern în PIO și transmisă către UC după revalidare cu $D_7=1$. Biții D_6, D_5 și D_4 sînt utilizați în general numai în cazul selectării modului 3 de lucru. Poziționarea pe „1" a lui D_4 într-un cuvînt de comandă-întreruperi, transmis în timp ce *port*-ul ales lucrează în oricare din cele patru moduri va conduce la ștergerea unei eventuale întreruperi în așteptare. Pentru precizarea condițiilor de funcționare ale *port*-ului selectat în modul 3, biții $D_6 \div D_4$ trebuie programați astfel: D_6 definește funcția logică ce conduce la generarea întreruperii, $D_6 = 1$ specifică funcția *ȘI* iar $D_6 = 0$ funcția *SAU*; D_5 precizează polaritatea activă a liniilor *port*-ului: $D_5=1$ înseamnă că liniile sînt active pe „1" iar $D_5=0$ pe „0"; $D_4=1$ înseamnă că următorul cuvînt transmis către PIO va fi un cuvînt de definire a măștilor, cuvînt ce are formatul următor:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
MB_7	MB_6	MB_5	MB_4	MB_3	MB_2	MB_1	MB_0

Numai liniile *port*-ului ai căror biți de mascare, MB_i, sînt zero, vor fi luate în considerație la generarea unei întreruperi.

Bistabilul de validare a întreruperilor unui *port* poate fi pus pe „1" sau pe „0" separat, fără a modifica acțiunea cuvîntului de comandă-întreruperi precedent, prin transmiterea unei comenzi cu formatul:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Validare întreruperi	X	X	X	0	0	1	1

Această comandă este, evident, asincronă cu generarea întreruperii de către *port*, generare care poate să apară chiar pe timpul cît UC transmite octetul de invalidare la PIO; *port*-ul generează o întrerupere achitată de unitatea centrală, dar vectorul nu se mai transmite pe durata ciclului de achitare datorită comenzii de invalidare. Ca rezultat va apărea o eroare de program. Pentru a rezolva această situație programatorul trebuie să invalideze temporar întreruperile pe durata transmiterii comenzii spre PIO-Z80, ca în secvența de mai jos:

```
    ...
    LD  A,03H       ; Încărcare A cu comanda de invalidare
    DI              ; Invalidare intreruperi UC
    OUT (PIO),A     ; Transmitere comandă către PIO
    EI              ; Validare întreruperi UC
    ...
```

1.2.4.4. INIȚIALIZAREA CIRCUITULUI

La punerea sub tensiune PIO-Z80 se inițializează după cum urmează:

— registrele pentru măști ale ambelor *port*-uri sînt puse pe „1", inhibîndu-se astfel toți biții;

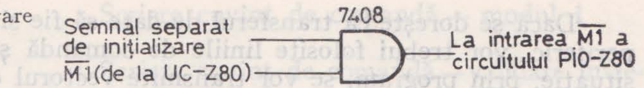


Fig. 1.35. Inițializarea hardware a lui PIO-Z80

— este selectat modul 1 de lucru, intrare octet, liniile $A_0 \div A_7$, $B_0 \div B_7$ fiind trecute în starea de impedanță mare, iar ieșirile ARDY și BRDY pe „0";

— registrele de ieșire ale celor două *port*-uri sînt puse pe „0";

— bistabilii de validare a întreruperilor din ambele *port*-uri sînt puși pe „0", invalidîndu-se astfel generarea întreruperilor; de subliniat că registrele care păstrează vectorii de întrerupere nu se inițializează.

În afară de această inițializare la punerea sub tensiune, PIO-Z80 mai poate fi inițializat prin activarea intrării $\overline{M1}$ cu condiția ca $\overline{RD}$ și $\overline{IORQ}$ să rămînă pe „1". După ce $\overline{M1}$ va trece pe „1" circuitul intră în starea inițială descrisă mai sus. Acest mod de inițializare hardware, fără linie separată de ștergere, a fost impus de limitările datorate necesității de a împacheta *chip*-ul într-o capsulă cu 40 de conexiuni externe. Astfel, se permite ștergerea circuitului cu ajutorul unui semnal separat de inițializare, activ pe „0", care comandă intrarea $\overline{M1}$ a lui PIO, împreună cu semnalul $\overline{M1}$ generat de UC-Z80 prin intermediul unei porți 7408 ca în figura 1.35.

PIO-Z80 rămîne în starea inițială pînă la recepționarea unui cuvînt de comandă de la UC.

1.2.4.5. PROGRAMAREA CIRCUITULUI PENTRU FUNCȚIONARE ÎN MODUL 0

În exemplele ce urmează se va presupune că circuitul PIO este selectat cu ajutorul bitului de adresă A_3, iar liniile de comandă C/D Sel și B/A Sel sînt acționate de biții cei mai puțin semnificativi ai magistralei de adresă; adresele de I/E utilizate sînt, în acest caz, cele date în tabelul de mai jos:

Conexiuni externe			Adrese de I/E	Semnificația octetului transferat
$\overline{CE}$ ($\overline{A_3}$)	C/D Sel(A_1)	B/A Sel(A_0)		
0	0	0	08H	Date *port* A
0	0	1	09H	Date *port* B
0	1	0	0AH	Comenzi *port* A
0	1	1	0BH	Comenzi *port* B

Pentru a folosi unul din *port*-urile circuitului PIO-Z80, de exemplu A, în modul 0, ieșire-octet, fără a utiliza liniile de comandă a transferului și întreruperea, este suficientă transmiterea către *port*-ul respectiv a modului și, apoi, a datelor. O secvență de program care realizează aceste lucruri este și cea dată în continuare:

```
...
LD   A,0FH   ; Încărcare A cu cuvîntul de comandă — mod 0
OUT (0AH),A ; Transmitere comenzi port A
LD   A,77H   ; Încărcare A cu octet-date
OUT (08H),A ; Transmitere date port A
...
```

Dacă se dorește ca transferul de date să fie sincronizat cu echipamentul periferic, vor trebui folosite liniile de comandă și întreruperea. În această situație, prin program, se vor transmite vectorul de întrerupere, modul, cuvîntul de comandă-întrerupere și se va inițializa semnalul READY; datele vor fi trimise în subrutina de serviciu a întreruperii:

```
MOD0:  IM  2            ; Programare Z80 în întreruperi, modul 2
       LD  HL, TAB      ; În HL adresa de început a tabelei locațiilor
       LD  A,H          ; unde se găsesc adresele subrutinelor de serviciu
       LD  I,A          ; În I octetul mai semnificativ al adresei lo-
                        ; cației
       LD  IX,SRVM0     ; Încărcare în tabelă a adresei subrutinei de
       LD  (TAB+06H),IX ; serviciu pentru PIO — modul 0
       LD  A,06H        ; Scriere vector de întrerupere
       OUT (0AH),A
       LD  A,0FH        ; Scriere cuvînt de comandă — modul 0
       OUT (0AH),A
       LD  A,87H        ; Scriere cuvînt de comandă — validare între-
       OUT (0AH), A     ; ruperi
       LD  A,0FFH       ; Inițializare semnal ARDY printr-o scriere
       OUT (08H),A      ; falsă a unui octet de date
       EI               ; Validare întreruperi Z80
       ...
SRVM0: ...              ; Subrutina de serviciu pentru PIO — modul 0
       PUSH AF          ; Salvare stare UC
       ...
       LD  A,(DATE)     ; Scriere octet-date
       OUT (08H),A
       ...
       POP AF           ; Refacere stare UC
       ...
       EI               ; Validare întreruperi
       RETI             ; Revenire din subrutina de serviciu
```

1.2.4.6. PROGRAMAREA CIRCUITULUI PENTRU FUNCȚIONARE ÎN MODUL 1

Dacă se dorește utilizarea unuia din *port*-urile circuitului PIO-Z80, în exemplul care urmează *port*-ul B, în modul 1, intrare-octet, prin program se vor trimite: vectorul de întrerupere, modul de lucru și cuvîntul de comandă pentru validarea întreruperilor; datele sînt preluate în subrutina de serviciu. Dăm în continuare o secvență de programare a *port*-ului B în modul 1.

```
MOD1:  IM  2            ; Programare Z80 în întreruperi, modul 2
       LD  HL,TAB       ; În HL adresa de început a tabelei locațiilor
       LD  A,H          ; unde se găsesc adresele subrutinelor de serviciu
       LD  I,A          ; În I octetul mai semnificativ al adresei locației
       LD  IX,SRVM1     ; Încărcare în tabelă a adresei subrutinei de
       LD  (TAB+08H),IX ; serviciu pentru PIO — modul 1
       LD  A,08H        ; Scriere vector de întrerupere
       OUT (0BH),A
```

```
         LD  A,4FH           ; Scriere cuvînt de comandă — modul 1
         OUT (0BH),A
         LD  A,87H           ; Scriere cuvînt de comandă — validare între-
         OUT (0BH),A         ; ruperi
         IN  A,(09H)         ; Inițializare BRDY printr-o citire falsă a
                             ; unui octet de date
         EI                  ; Validare întreruperi Z80
         ...
SRVM1:   ...                 ; Subrutina de serviciu pentru PIO — modul 1
         PUSH AF             ; Salvare stare UC
         ...
         IN   A,(09H)        ; Citire octet-date
         ...
         POP  AF             ; Refacere stare UC
         ...
         EI                  ; Validare întreruperi
         RETI                ; Revenire din subrutina de serviciu
```

1.2.4.7. PROGRAMAREA CIRCUITULUI PENTRU FUNCȚIONARE ÎN MODUL 2

Modul 2, bidirecțional pe octet, poate fi programat numai pentru *port*-ul A, utilizînd semnalele de comandă ale ambelor *port*-uri. Semnalele de comandă ale *port*-ului A sînt folosite în operațiile de ieșire-date, iar cele ale *port*-ului B în intrări-date. Pentru ca *port*-ul A să lucreze în modul 2, *port*-ul B va trebui programat în modul 3. În continuare se dă o secvență de programare a *port*-ului A în modul 2.

```
MOD2:  IM  2               ; Programare Z80 în întreruperi, modul 2
       LD  HL, TAB         ; În HL adresa de început a tabelei locațiilor
       LD  A,H             ; unde se găsesc adresele subrutinelor de serviciu
       LD  I,A             ; În I octetul mai semnificativ al adresei locației
       LD  IX,SRVM0        ; Încărcare adresă subrutină serviciu ieșire-
       LD (TAB+06H),IX     ; date (aceeași ca la modul 0)
       LD  IX,SRVM1        ; Încărcare adresă subrutină serviciu — intrare-
       LD (TAB+08H),IX     ; date (aceeași ca la modul 1)
       LD  A,06H           ; Scriere vector-întrerupere pentru port-ul A
       OUT (0AH),A
       LD  A,08H           ; Scriere vector-întrerupere pentru port-ul B
       OUT (0BH),A
       LD  A,8FH           ; Comandă mod 2 pentru port-ul A
       OUT (0AH),A
       LD  A,0CFH          ; Comandă mod 3 pentru port-ul B
       OUT (0BH),A
       LD  A,0FFH          ; Liniile port-ului B definite ca intrări
       OUT (0BH),A         ; (după inițializare toți biții sînt mascați)
       LD  A,87H           ; Validare întreruperi
       OUT (0AH),A         ; pentru port-ul A
       OUT (0BH),A         ; pentru port-ul B
       LD  A,0FFH          ; Inițializare ARDY
       OUT (08H),A
```

```
       IN   A,(09H)        ; Inițializare BRDY
       EI                  ; Validare întreruperi
       ...
```

1.2.4.8. PROGRAMAREA CIRCUITULUI PENTRU FUNCȚIONARE ÎN MODUL 3

În modul 3 pot fi programate să lucreze ambele *port*-uri ale unui circuit PIO-Z80. Acest mod de funcționare, numit și mod de comandă, realizează operații de intrare-ieșire pe bit, fără să utilizeze liniile READY și STROBE. Secvența care urmează programează *port*-ul B în modul 3.

```
MOD3:  IM   2              ; Programare Z80 în întreruperi, modul 2
       LD   HL, TAB        ; În HL adresa de început a tabelei locațiilor
       LD   A, H           ; unde se găsesc adresele subrutinelor de serviciu
       LD   I,A            ; În I octetul mai semnificativ al adresei locației
       LD   IX,SRVM3       ; Încărcare adresă subrutină serviciu (mod 3)
       LD (TAB+0CH),IX
       LD A, 0CH           ; Scriere vector-întrerupere pentru port-ul B
       OUT (0BH),A
       LD   A,0CFH         ; Comandă mod 3 pentru port-ul B
       OUT (0BH),A
       LD   A, 0FH         ; Definire linii de intrare pentru port-ul B
       OUT (0BH),A
       LD   A,97H          ; Scriere cuvînt de comandă întreruperi
       OUT (0BH),A
       LD   A,0FCH         ; Mascare toate liniile în afară de B0 și B1
       OUT (0BH),A
       LD   A,00H          ; Inițializare linii de ieșire pe zero
       OUT (09H),A
       EI                  ; Validare întreruperi
       ...
SRVM3: ...                 ; Subrutina de serviciu pentru modul 3
       PUSH AF             ; Salvare stare UC
       ...,
       IN A,(09H)          ; Citire linii intrare port B
       AND 0FH
       LD   (DISP),A       ; Afișare linii de intrare
       CALL AFIS
       RLA                 ; Transmitere linii de intrare pe liniile de
       RLA                 ; ieșire ale port-ului B
       RLA
       RLA
       OUT (09H),A
       ...
       POP AF              ; Refacere stare UC
       ...
       EI                  ; Validare întreruperi
       RETI                ; Revenire din subrutina de serviciu
```

Modul 3 oferă o mare flexibilitate în definirea funcționării unui *port*. De aceea mai multe opțiuni pot fi selectate și specificate prin intermediul octeților de comandă. În exemplul de mai sus prin progam se execută următoarele operații:

— programarea UC-Z80 în modul 2 de lucru cu întreruperile;

— inițializarea registrului I;

— încărcarea adresei subrutinei de serviciu specifice în tabela vectorilor de întrerupere;

— scrierea vectorului de întrerupere pentru *port*-ul B;

— programarea *port*-ului B în modul 3;

— definirea registrului de selecție I/E: 1 — linie de intrare, 0 — linie de ieșire;

— scrierea cuvîntului de comandă-întreruperi: validare-întreruperi, selecția funcției SAU pentru generarea întreruperii (activarea cel puțin a unei linii nemascate a *port*-ului conduce la generarea întreruperii), selecția nivelului activ „0", specificarea faptului că urmează un cuvînt de măști;

— transmiterea cuvîntului de măști: aici *port*-ul B e programat să genereze o întrerupere numai dacă unul din biții B_0, B_1 devine „0".

Subrutina de serviciu SRVM3, scrisă ca exemplu, realizează următoarele funcțiuni:

— salvează starea UC;

— citește cele patru linii de intrare ale *port*-ului B;

— afișează aceste linii asamblate într-un octet a cărui parte semnificativă este zero (scriere la locația DISP, apel subrutină AFIS);

— scrie în biții de ieșire ai *port*-ului B valoarea citită mai sus a liniilor de intrare ale aceluiași *port*;

— reface starea UC;

— validează întreruperile mascabile;

— redă comanda programului principal.

1.3. CIRCUITUL NUMĂRĂTOR-TEMPORIZATOR CTC-Z80

Circuitul numărător-temporizator CTC-Z80 face parte din familia Z80 și este destinat implementării funcțiilor de numărare și măsurare a timpului. CTC poate realiza aceste funcții pe patru canale independente de 8 biți, interfațîndu-se direct cu magistralele de date și comenzi ale microprocesorului Z80. Circuitul poate fi programat software, astfel încît fiecare canal să lucreze independent într-unul din cele două moduri: ca numărător sau ca temporizator. În *modul numărător* CTC-Z80 numără impulsuri din exteriorul sistemului și, dacă a fost programat să lucreze cu întreruperile, generează o întrerupere spre UC după un număr prestabilit de impulsuri. Ca și PIO-Z80, CTC funcționează în întreruperi conform modului 2 de lucru al UC-Z80. Vectorul și numărul de impulsuri ce trebuie numărate pînă la generarea întreruperii se pot transmite software. În *modul temporizator* CTC numără impulsurile ceasului de sistem, Φ. Generînd, ca și în celălalt mod, o întrerupere după un număr prestabilit de impulsuri, circuitul asigură măsurarea precisă a unor intervale de timp definite, măsurare necesară, de exemplu, în prelucrările în timp real.

Circuitul poate fi programat ușor prin transmiterea a doi octeți pentru fiecare canal; validarea întreruperilor se face cu ajutorul unui al treilea. Odată pornit, canalul decrementează, își reîncarcă la sfîrșitul numărării, în mod automat, constanta de timp și își reia numărarea. Numărarea ceasului Φ sau a evenimentelor externe se face pe front pozitiv sau negativ, ales prin software. Programarea întreruperilor e simplificată, fiind necesară transmiterea unui singur octet, circuitul generînd un vector unic pentru fiecare canal prin modificarea corespunzătoare a biților 1 și 2.

1.3.1. STRUCTURA CIRCUITULUI CTC-Z80

În figura 1.36 se prezintă structura circuitului CTC, la nivel de schemă bloc. După cum se vede, el e organizat în jurul unei magistrale interne, fiind compus dintr-un bloc de logică pentru interfața cu UC-Z80, un bloc destinat funcției globale de comandă internă, logica de comandă a întreruperilor, cele patru canale pentru numărare sau măsurarea timpului.

Blocul de logică pentru interfațarea cu UC-Z80 decodifică intrările de adresă și distribuie semnalele de interfață pe magistrala internă a circuitului. Logica de comandă internă controlează funcționarea globală a circuitului prin semnale generale care afectează operații cum sînt: selecția circuitului, inițializarea sau logica de scriere/citire. Logica de comandă a întreruperilor asigură respectarea procedurii de cerere-achitare întrerupere conform modului 2 de lucru al microprocesorului Z80. De asemenea, această parte a circuitului controlează funcționarea într-un lanț de priorități, cu ajutorul semnalelor IEI și IEO. Organizarea unui canal de numărare/măsurare a timpului este dată

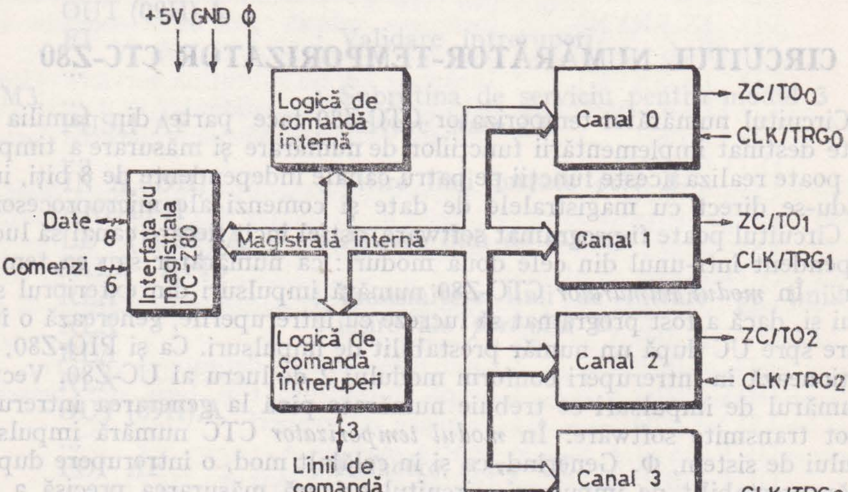
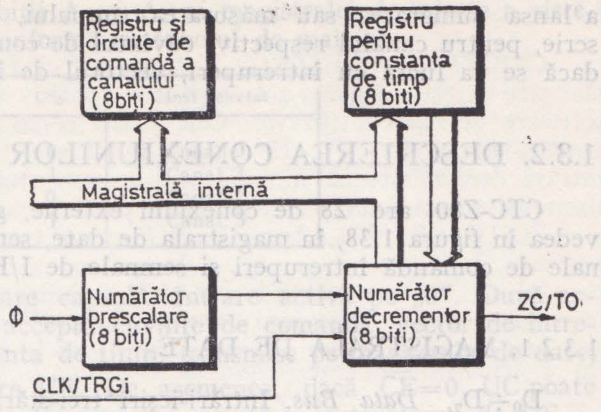


Fig. 1.36. Schema bloc a circuitului CTC-Z80

Fig. 1.37. Organizarea unui canal de numărare/măsurare a timpului



în figura 1.37. El este alcătuit din două registre, două numărătoare şi logica de comandă specifică. La programare, registrul de comandă se încarcă cu un cuvînt de comandă, logica asociată stabilind apoi funcţionarea în detaliu a canalului respectiv prin decodificarea acestui cuvînt şi precizarea următoarelor condiţii:

— validare/invalidare întreruperi;
— funcţionare în modul numărător sau în modul temporizator;
— factor de prescalare la măsurarea timpului (16 sau 256);
— frontul activ al intrării CLK/TRG;
— declanşare automată sau cu ajutorul intrării CLK/TRG în modul temporizator;
— urmează transmiterea constantei de timp;
— iniţializare software.

Registrul pentru constanţa de timp memorează o valoare de 8 biţi (între 1 şi 256, 0 — reprezentînd 256) ce va fi încărcată automat în numărătorul-decrementor la iniţializare şi, apoi, la fiecare trecere prin zero.

Numărătorul de prescalare, utilizat numai în modul de măsurare a timpului, divide frecvenţa ceasului de sistem cu 16 sau 256. Ieşirea acestui numărător constituie intrare de ceas pentru numărătorul-decrementor. Acest din urmă numărător poate fi deci decrementat, fie de ieşirea numărătorului de prescalare, în modul de măsurare a timpului, fie de intrarea CLK/TRG, în modul de numărare.

UC-Z80 poate citi în orice moment, fără a perturba funcţionarea circuitului, prin execuţia unei instrucţiuni de I/E, numărul rămas de numărat pînă la zero. La trecerea prin zero ieşirea ZC/TO va fi pulsată şi, dacă întreruperile sînt validate, se va face o cerere de întrerupere la unitatea centrală.

După punerea sub tensiune CTC-Z80 rămîne într-o stare nedefinită. Activarea intrării $\overline{\text{RESET}}$ va conduce la iniţializarea circuitului dar, pentru

a lansa numărarea sau măsurarea timpului, programatorul va trebui să scrie, pentru canalul respectiv, cuvîntul de comandă, constanta de timp și, dacă se va lucra cu întreruperi, vectorul de întrerupere.

1.3.2. DESCRIEREA CONEXIUNILOR EXTERNE

CTC-Z80 are 28 de conexiuni externe, grupate, după cum se poate vedea în figura 1.38, în magistrala de date, semnale de comandă UC, semnale de comandă întreruperi și semnale de I/E pe canale.

1.3.2.1. MAGISTRALA DE DATE

$D_0 \div D_7$, *Data Bus*. Intrări-ieșiri trei-stări active pe „1". Magistrala se utilizează pentru transferul datelor și comenzilor între UC și CTC-Z80.

1.3.2.2. SEMNALELE DE COMANDĂ UC

CS_0, CS_1, *Channel Select*, selecție canal. Intrări active pe „1", folosite pentru selecția unuia din cele patru canale independente ale circuitului.

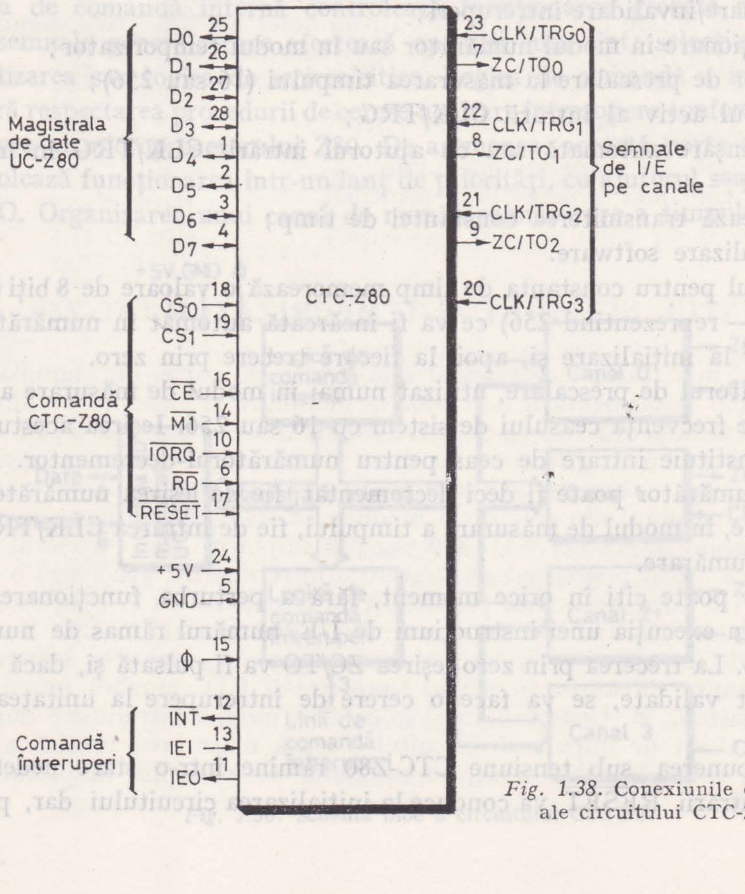


Fig. 1.38. Conexiunile externe ale circuitului CTC-Z80

De obicei se conectează la biţii A_0 şi A_1 ai magistralei de adrese a sistemului. Selecţia canalului e conformă cu tabelul de mai jos:

CS_1	CS_0	Canal selectat
0	0	Canal 0
0	1	Canal 1
1	0	Canal 2
1	1	Canal 3

$\overline{CE}$, *Chip Enable,* validare capsulă. Intrare activă pe „0". După activarea acestei intrări, CTC acceptă cuvinte de comandă, vector de întrerupere sau cuvinte cu constanta de timp, transmise pe magistrala de date, în timpul unor cicli de scriere I/E. De asemenea, dacă $\overline{CE}=0$, UC poate citi conţinutul numărătorului-decrementor printr-o instrucţiune de I/E.

$\overline{M1}$, *Machine Cycle One,* primul ciclu-maşină. Intrare de la UC-Z80 activă pe „0". Împreună cu $\overline{RD}$ semnifică un ciclu de extragere iar împreună cu $\overline{IORQ}$ un ciclu de achitare a întreruperii.

$\overline{IORQ}$, *Input/Output Request,* cerere de I/E. Intrare de la UC-Z80, activă pe „0". În timpul unui ciclu de scriere, $\overline{IORQ}$ şi $\overline{CE}$ sînt active, iar $\overline{RD}$ inactivă: CTC-Z80, ca şi PIO-Z80, nu primeşte un semnal de comandă-scriere separat, acesta generîndu-se intern prin inversarea lui $\overline{RD}$. La un ciclu de citire, $\overline{IORQ}$, $\overline{CE}$ şi $\overline{RD}$ sînt active, conţinutul numărătorului-decrementor fiind plasat pe magistrala de date către UC. $\overline{IORQ}$ şi $\overline{M1}$ active împreună se utilizează în ciclii de achitare a întreruperii, cînd canalul prioritar îşi plasează vectorul de întrerupere pe $D_0 \div D_7$.

$\overline{RD}$, *Read Cycle Status,* ciclu de citire. Intrare de la UC-Z80 activă pe „0". Semnal utilizat împreună cu $\overline{IORQ}$ şi $\overline{CE}$ în operaţiile de transfer de date şi comenzi între UC şi CTC.

1.3.2.3. SEMNALELE DE COMANDĂ A ÎNTRERUPERILOR

IEI, *Interrupt Enable In,* intrare validare întreruperi. Intrare activă pe „1"; activă semnifică inexistenţa unor cereri de întrerupere prioritare în curs de tratare de către UC-Z80.

IEO, *Interrupt Enable Out,* ieşire validare întreruperi. Ieşire activă pe „1" numai cînd IEI = 1 şi UC-Z80 nu tratează nici-o întrerupere generată de vreunul din canalele de numărare/măsurare ale circuitului.

$\overline{INT}$, *Interrupt Request,* cerere întrerupere. Ieşire de tip drenă-în-gol activă pe „0". Se activează cînd oricare din canalele CTC căruia i s-a validat prin program generarea întreruperii trece prin zero.

$\overline{RESET}$, iniţializare. Intrare activă pe „0" utilizată pentru iniţializarea hardware, a circuitului. Acest semnal opreşte numărarea în toate canalele, invalidează generarea întreruperilor, trece ieşirile ZC/TO şi $\overline{INT}$ în starea inactivă, face IEO=IEI şi trece circuitele de comandă a magistralei de date în starea de impedanţă mare.

1.3.2.4. SEMNALELE DE I/E PE CANALE

$CLK/TRG_0 \div CLK/TRG_3$, *External Clock/Timer Trigger*, ceas extern/declanșare externă. Intrări active pe front pozitiv sau negativ, la alegere prin software de către utilizator. Cele patru conexiuni externe corespund celor patru canale ale CTC. În modul numărare fiecare front activ al acestor intrări decrementează numărătorul-decrementor. În modul de măsurare a timpului frontul activ este utilizat pentru începerea măsurării.

$ZC/TO_0 \div ZC/TO_2$, *Zero Count/Timeout*, trecere prin zero/terminare temporizare. Ieșire activă pe „0". Cele trei conexiuni corespund canalelor $0 \div 2$, canalul al treilea neavînd o astfel de ieșire datorită limitelor impuse de capsula cu 28 de conexiuni externe. În ambele moduri la aceste ieșiri sînt generate impulsuri pozitive cînd numărătoarele-decrementoare trec prin zero.

1.3.3. DIAGRAME DE TIMP

1.3.3.1. CICLUL DE SCRIERE UC

Cuvîntul de comandă, vectorul de întrerupere sau constanta de timp se transmit conform diagramei de timp din figura 1.39. CTC-Z80 nu are o intrare separată pentru comanda de scriere: acest semnal este generat intern dacă intrarea RD e „1" pe durata stării T_1. În timpul stării T_2 $\overline{IORQ}$ și $\overline{CE}$ sînt pe „0" iar $\overline{M1}$ pe „1" pentru a distinge ciclul de scriere de ciclul de achitare a întreruperii. Valoarea intrărilor CS_0 și CS_1 selectează canalul unde se vor scrie datele prezente pe magistrală, cu frontul pozitiv al ceasului din starea T_3.

1.3.3.2. CICLUL DE CITIRE UC

Ciclul de citire, a cărui desfășurare în timp este prezentată în figura 1.40, se utilizează pentru citirea conținutului numărătorului-decrementor, citire efectuată fără perturbarea decrementării. În timpul stării T_2, UC-Z80 inițiază un ciclu de citire activînd intrările RD, IORQ și CE ale CTC-ului. Valoarea intrărilor CS_0 și CS_1 selectează canalul, care va plasa, cu frontul pozitiv al ceasului din T_3, valoarea numărătorului-decrementor pe magistrala de date a sistemului.

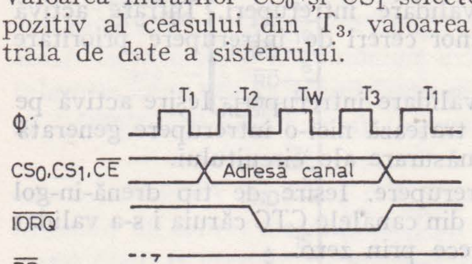


Fig. 1.39. Ciclul de scriere UC în CTC-Z80

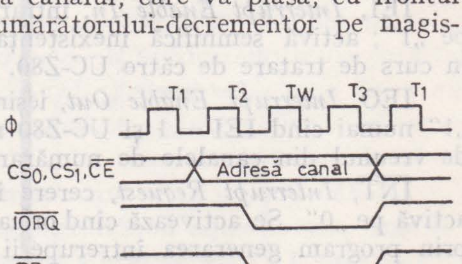


Fig. 1.40. Ciclul de citire UC din CTC-Z80

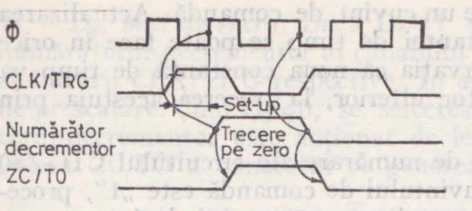


Fig. 1.41. Funcționarea unui canal în modul numărător

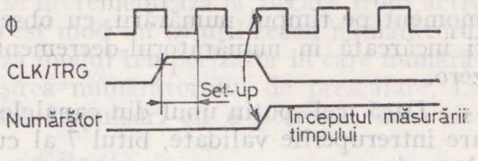


Fig. 1.42. Funcționarea unui canal în modul temporizator

1.3.3.3. MODUL NUMĂRĂTOR

În acest mod de lucru decrementarea numărătorului-decrementor se face, așa cum se poate vedea în figura 1.41, cu fiecare front activ al intrării CLK/TRG (aici frontul pozitiv), sincron cu ceasul Φ. Pentru ca decrementarea să apară cu primul front pozitiv al lui Φ, care urmează frontului activ al lui CLK/TRG, trebuie respectat un timp de *set-up*, de stabilizare, precizat în catalog, de exemplu 300 ns pentru versiunea CTC-Z80 fabricată de Zilog (frecvența ceasului Φ de 2,5 MHz). Dacă nu se respectă acest timp, numărarea va fi întîrziată cu o perioadă a ceasului Φ. De asemenea, impulsul de declanșare, în figura 1.41 pozitiv, este necesar să aibă o durată minimă, iar perioada semnalului CLK/TRG să fie cel puțin de două ori mai mare decît perioada ceasului Φ. Ieșirea ZC/TO corespunzătoare va fi activată la trecerea pe zero a numărătorului-decrementor.

1.3.3.4. MODUL TEMPORIZATOR

Figura 1.42 ilustrează desfășurarea în timp a funcționării unui canal programat să lucreze în modul temporizator. Măsurarea timpului se declanșează cu al doilea front pozitiv al ceasului Φ ce urmează frontului activ al intrării CLK/TRG, aici pozitiv. Ca și mai sus, pentru respectarea acestei secvențe de timp vor trebui îndeplinite condițiile unui timp de *set-up* între frontul activ al intrării CLK/TRG și primul front pozitiv al lui Φ și, de asemenea, cea a unei durate minime a impulsului de declanșare a măsurării timpului. Dacă frontul activ al intrării CLK/TRG apare mai tîrziu decît timpul de *set-up*, inițierea numărării va fi întîrziată cu o perioadă de ceas.

Precizăm că măsurarea timpului poate fi declanșată și automat, fără utilizarea conexiunii externe CLK/TRG, prin transmiterea unei anumite configurații a cuvîntului de comandă.

1.3.4. PROGRAMAREA CIRCUITULUI CTC-Z80

Pentru a programa circuitul CTC-Z80 se transmit cuvîntul de comandă, constanta de numărare și, dacă se dorește utilizarea întreruperilor, vectorul de întrerupere. Cuvîntul de comandă este folosit pentru selectarea modului de lucru și a altor parametri ce vor caracteriza funcționarea canalului programat. Constanta de timp, o valoare binară cuprinsă în intervalul

(1, 256), este întotdeauna precedată de un cuvînt de comandă. Actualizarea cuvîntului de comandă și/sau a constantei de timp se poate face în orice moment pe timpul numărării, cu observația că noua constantă de timp va fi încărcată în numărătorul-decrementor ulterior, la trecerea acestuia prin zero.

Dacă cel puțin unul din canalele de numărare ale circuitului CTC-Z80 are întreruperile validate, bitul 7 al cuvîntului de comandă este „1", procedura de programare va cuprinde și transmiterea vectorului de întrerupere. Este necesară scrierea unui singur vector pentru întregul circuit deoarece logica de comandă a întreruperilor modifică în mod automat vectorul plasat pe magistrală de către CTC, în ciclul de achitare, precizînd canalul care solicită întreruperea.

Selecția canalului programat se face cu ajutorul intrărilor CS_0 și CS_1 conform tabelului dat în § 1.3.2.

1.3.4.1. CUVÎNTUL DE COMANDĂ

Pentru a încărca un cuvînt de comandă, unitatea centrală execută o instrucțiune de I/E la adresa corespunzătoare canalului ales. Octetul transmis va fi interpretat drept cuvînt de comandă și înscris în registrul de comandă al canalului respectiv dacă bitul 0, cel mai puțin semnificativ, este „1". Cu ajutorul celorlalți șapte biți se selectează modul de lucru și parametrii de funcționare. Formatul cuvîntului de comandă este următorul:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Validare întreruperi	Mod de lucru	Factor prescalare	Front activ	Mod de declanșare	Urmează constanta de timp	Inițializare	1

D_7, *validare-întreruperi*. Dacă $D_7=1$, canalul respectiv este validat să genereze o întrerupere la fiecare trecere prin zero a numărătorului-decrementor. În această situație, se încarcă în CTC și vectorul de întrerupere. Întreruperile pot fi validate în oricare din cele două moduri de lucru. Dacă se trimite un cuvînt de comandă de actualizare la un canal în funcționare, cuvînt avînd $D_7=1$, validarea întreruperilor nu va fi retroactivă în raport cu o posibilă trecere anterioară a numărătorului prin zero. $D_7=0$ înseamnă invalidarea întreruperilor, ștergerea întreruperii în așteptare pe canalul respectiv. Ca și la PIO poate apărea un asincronism între generarea întreruperii de către CTC și invalidarea ei de către UC, situație care, datorită citirii unui vector fals de întrerupere, ar conduce la o eroare de program. Pentru a rezolva această problemă se poate utiliza următoarea secvență de program:

```
...
LD A,01H        ; Încărcare A cu comanda de invalidare
DI              ; Invalidare întreruperi UC
OUT (CTC),A     ; Invalidare întreruperi CTC
EI              ; Validare întreruperi UC
...
```

D_6, *mod de lucru*. Dacă $D_6=1$ este selectat modul numărător în care numărătorul-decrementor al canalului se decrementează la fiecare front activ al intrării CLK/TRG respective. În acest mod nu se utilizează numărătorul de prescalare. Cînd $D_6=0$, se selectează modul temporizator în care numărătorul-decrementor este acționat de ieșirea numărătorului de prescalare. La ieșirea ZC/TO a canalului se generează impulsuri cu o perioadă dată de:

$$T_{ZC/TO}=t_{\Phi}\cdot FP\cdot CT$$

unde t_{Φ} este perioada ceasului de sistem, Φ, *FP* — factorul de prescalare, 16 sau 256, iar *CT* — constanta de timp încărcată în registrul pentru constanta de timp al canalului respectiv.

D_5, *factor-prescalare*. Bitul D_5, definit numai pentru modul temporizator, precizează cu cît trebuie să se dividă frecvența ceasului Φ: cu 16, $D_5=0$, sau cu 256, $D_5=1$.

D_4, *front activ*. Acest bit stabilește care din fronturile impulsului CLK/TRG va fi utilizat pentru lansarea măsurării timpului în modul temporizator, respectiv pentru decrementare în modul numărător. Frontul pozitiv este selectat cu $D_4=1$ iar cel negativ cu $D_4=0$. O reprogramare a frontului activ pe durata numărării echivalează cu un front activ, conducînd la decrementarea numărătorului-decrementor. De asemenea, o schimbare a frontului activ, printr-un cuvînt de comandă de actualizare, în timp ce canalul așteaptă să înceapă măsurarea timpului, conduce la startarea numărătoarelor, echivalînd deci cu un impuls de declanșare.

D_3, *mod de declanșare*. Definit numai pentru modul temporizator, acest bit specifică modul de declanșare, de începere a măsurării timpului: $D_3=1$, declanșare externă cu ajutorul intrării CLK/TRG (vezi figura 1.42); $D_3=0$, declanșare internă, automată, cînd începerea măsurării timpului se face pe frontul pozitiv al ceasului Φ în starea T_2 din ciclul-mașină următor celui în care s-a scris constanta de timp. După declanșare, numărarea se face în mod continuu, la trecerea prin zero constantele încărcîndu-se automat, numărătoarele fiind decrementate fără întrerupere sau întîrziere pînă la o inițializare hardware sau software.

D_2, *urmează constanta de timp*. $D_2=1$ indică faptul că următorul cuvînt scris la adresa canalului programat este o constantă de timp, ce poate fi transmisă în orice moment, numărătorul-decrementor continuînd decrementarea constantei anterioare pînă la trecerea prin zero, după care o va încărca pe cea nouă. $D_2=0$ înseamnă că nu urmează o constantă. Cuvîntul de comandă transmis cu $D_2=0$ este de obicei un cuvînt de actualizare a registrului de comandă asociat canalului deja în funcțiune. Aceasta deoarece un canal nu poate funcționa fără să i se fi transmis în mod corect constanta de timp și singurul mod de a o transmite este prin poziționarea pe „1" a bitului D_2, într-un cuvînt de comandă anterior.

D_1, *inițializare*. Poziționarea pe „1" a acestui bit conduce la o inițializare software, canalul oprindu-se din numărare sau din măsurarea timpului. Scrierea în D_1 a unui impuls de inițializare oprește funcționarea canalului, dar nu modifică nici unul din biții registrului de comandă. Dacă $D_2=D_1=1$, canalul își va relua funcționarea după încărcarea constantei de timp. Dacă $D_1=0$, canalul își continuă funcționarea curentă.

1.3.4.2. CONSTANTA DE TIMP

Pentru a începe numărarea în oricare din cele două moduri de lucru, un canal CTC trebuie să primească o constantă de timp. Constanta de timp poate avea orice valoare întreagă cuprinsă între 1 şi 256, cu 00H interpretat ca 256. Încărcarea unei constante se face în doi paşi: întîi transmiterea unui cuvînt de comandă cu bitul $D_2=1$, apoi scrierea unui octet avînd valoarea constantei. Transmiterea unei constante spre un canal aflat în funcţiune va conduce la încărcarea ei în registrul pentru constantă, de unde va fi transferată în numărătorul-decrementor, la terminarea operaţiei curente, adică la trecerea acestuia prin zero.

1.3.4.3. VECTORUL DE ÎNTRERUPERE

Vectorul de întrerupere se transmite dacă cel puţin unul din canalele CTC are întreruperile validate. CTC-Z80 lucrează în întreruperi la fel ca PIO-Z80, respectînd procedura corespunzătoare modulului 2 al microprocesorului Z80 (vezi § 1.1, 1.2). Programarea vectorului constă în scrierea unui octet în canalul 0 al CTC. Acest octet trebuie să aibă bitul cel mai puţin semnificativ $D_0=0$, pentru a fi distins de un cuvînt de comandă. Utilizatorul precizează în acest octet cei mai semnificativi cinci biţi, logica de comandă din CTC urmînd a stabili valoarea biţilor D_2 şi D_1 în funcţie de canalul care solicită întreruperea. Formatul vectorului de întrerupere este deci următorul:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
V_7	V_6	V_5	V_4	V_3	X	X	0

Se precizează de către utilizator (D_7–D_3)

Identificator de canal generat de CTC-Z80 (D_2, D_1):
00 = canal 0
01 = canal 1
10 = canal 2
11 = canal 3

Canalul 0 e prioritar faţă de celelalte canale.

1.3.4.4. PROGRAMAREA CIRCUITULUI PENTRU FUNCŢIONARE ÎN MODUL NUMĂRĂTOR

În exemplele următoare se va considera că circuitul CTC-Z80 este selectat cu ajutorul bitului de adresă A_4, iar liniile CS_0, CS_1 sînt comandate de biţii cei mai puţin semnificativi ai magistralei de adrese; în consecinţă adresele de I/E corespunzătoare celor patru canale vor fi cele din tabelul de mai jos:

Conexiuni externe			Adrese de I/E	Canal selectat
$\overline{CE}(A_4)$	$CS_1(A_1)$	$CS_0(A_0)$		
0	0	0	10H	Canal 0
0	0	1	11H	Canal 1
0	1	0	12H	Canal 2
0	1	1	13H	Canal 3

Canalul programat să lucreze în modul numărător își decrementează numărătorul-decrementor la fiecare front activ al intrării externe CLK/TRG. Dacă întreruperile sînt validate, la trecerea prin zero va fi generată o întrerupere, decrementarea reluîndu-se după încărcarea constantei păstrate în registrul pentru constanta de timp. Se dă în continuare o secvență de programare a canalului 2 în modul numărător.

```
MDNUM: IM  2             ; Programare Z80 în întreruperi, modul 2
       LD  HL,TAB        ; În HL adresa de început a tabelei locațiilor
       LD  A,H           ; unde se găsesc adresele subrutinelor de serviciu
       LD  I,A           ; În I octetul mai semnificativ al adresei locației
       LD  IX,SRVC2      ; Încărcare în tabelă a adresei subrutinei de
       LD  (TAB+1AH),IX; serviciu pentru CTC — canalul 2,
                         ; modul numărător
       LD  A,18H         ; Scriere vector de întrerupere la canalul 0
       OUT (10H),A
       LD  A,0C7H        ; Scriere cuvînt de comandă canal 2
       OUT (12H),A
       LD  A,(CONST)     ; Scriere constantă de timp canal 2
       OUT (12H),A
       EI                ; Validare întreruperi
       ...
```

Secvența MDNUM programează canalul 2 al CTC pentru a funcționa în modul numărător cu întreruperile validate. În acest scop secvența trebuie să precizeze conținutul a trei din registrele circuitului: registrul pentru vectorul de întrerupere, registrul de comandă al canalului 2 și registrul pentru constanta de timp, de asemenea al canalului 2.

Vectorul de întrerupere, scris întotdeauna la adresa de I/E a canalului 0, are cei mai semnificativi cinci biți poziționați prin program, ceilalți trei biți fiind determinați de CTC. Semnificația cuvîntului de comandă, 0C7H, recunoscut datorită faptului că are $D_0=1$, este următoarea:

$D_7=1$ validează întreruperile generate de canalele CTC. În cazul nostru trecerea prin zero a numărătorului-decrementor din canalul 2 va conduce la activarea liniei INT;

$D_6=1$ programează canalul 2 să lucreze în modul numărător;

$D_5=0$ neutilizat în modul numărător;

$D_4=0$ specifică frontul negativ al intrării CLK/TRG_2, ca front activ pe care se decrementează numărătorul;

$D_3=0$ neutilizat în modul numărător;

$D_2=1$ înseamnă că urmează scrierea constantei de timp;

$D_1=1$ specifică faptul că CTC — canalul 2 va începe să funcționeze, să numere, după încărcarea constantei de timp.

După execuția secvenței MDNUM, numărătorul-decrementor al canalului 2 va fi decrementat cu o unitate la fiecare front negativ apărut la intrarea CLK/TRG_2. După un număr de impulsuri, egal cu constanta de timp, CTC va genera o întrerupere activînd linia INT. În același timp, constanta va fi reîncărcată în numărătorul-decrementor contorizîndu-se în continuare fronturile negative ale semnalului CLK/TRG_2. Dacă subrutina de serviciu a întreruperii durează mai mult decît decrementarea constantei de timp, se pot pierde întreruperi, deoarece circuitul CTC-Z80 nu memorează o între-

rupere în aşteptare pe timpul cît întreruperea precedentă, generată de acelaşi canal, se află în curs de tratare. În această situaţie este asigurată numai corectitudinea funcţiei de numărare.

1.3.4.5. PROGRAMAREA CIRCUITULUI PENTRU FUNCŢIONARE ÎN MODUL TEMPORIZATOR

Vom ilustra programarea circuitului CTC-Z80 în modul temporizator printr-un exemplu de realizare a unui cronometru. Pentru aceasta, canalele 0, 1, 2 se conectează în cascadă, ieşirile ZC/TO_0 şi ZC/TO_1 legîndu-se la intrările CLK/TRG_1, respectiv CLK/TRG_2. Canalul 0 va lucra în modul temporizator iar canalele 1 şi 2 în modul numărător. Pornirea externă a cronometrului, a măsurării timpului, se face cu ajutorul unui impuls la intrarea CLK/TRG_0.

Numărul de secunde trecute, maximum 255, se înregistrează în numărătorul-decrementor al canalului 2. Pentru citirea numărului de secunde, cu ajutorul unei subrutine de serviciu, se utilizează canalul 3, programat în modul numărător. Acesta va genera o întrerupere la orice cerere externă de sfîrşit al cronometrării, cerere materializată printr-un impuls la intrarea CLK/TRG_3. Secvenţa de programare a celor patru canale ale CTC se dă mai jos:

```
MODT: IM   2               ; Programare Z80 în întreruperi, modul 2
      LD   HL,TAB          ; În HL adresa de început a tabelei locaţiilor
      LD   A,H             ; unde se găsesc adresele subrutinelor de serviciu
      LD   I,A             ; În I octetul mai semnificativ al adresei
                           ; locaţiei
      LD   IX,SRVCR        ; Încărcare în tabelă a adresei subrutinei de
      LD   (TAB+26H),IX    ; serviciu pentru CTC — cronometru (vector
                           ; generat de canalul 3)
      LD   A,26H           ; Scriere vector întrerupere la canalul 0
      OUT  (10H),A
      LD   A,2FH           ; Cuvînt de comandă canal 0
      OUT  (10H),A
      LD   A,98H           ; Constanta de timp pentru canalul 0
      OUT  (10H),A
      LD   A,47H           ; Cuvînt de comandă canal 1
      OUT  (11H),A
      LD   A,40H           ; Constanta de timp pentru canalul 1
      OUT  (11H),A
      LD   A,47H           ; Cuvînt de comandă canal 2
      OUT  (12H),A         ;
      LD   A,00H           ; Constanta de timp pentru canalul 2
      OUT  (12H),A
      LD   A,0C7H          ; Cuvînt de comandă canal 3
      OUT  (13H),A
      LD   A,01H           ; Constanta de timp pentru canalul 3
      OUT  (13H),A
      EI                   ; Validare întreruperi
      ...
```

```
SRVCR: ...          ; Subrutina de serviciu pentru CTC — cro-
                    ; nometru
       LD  C,12H    ; Citire canal 2 (număr secunde trecute)
       IN  B,(C)
       XOR A        ; Ştergere A
       SUB B        ; În A numărul de secunde trecute
       LD  (DATE),A ; Afişarea numărului de secunde
       CALL AFIS
       LD  A,2FH    ; Reiniţializare cronometru
       OUT (10H),A  ; Cuvînt de comandă canal 0
       LD  A,98H    ; Constanta de timp pentru canalul 0
       OUT (10H),A
       LD  A,47H    ; Cuvînt de comandă canal 1
       OUT (11H),A
       LD  A,40H    ; Constanta de timp pentru canalul 1
       OUT (11H),A
       LD  A,47H    ; Cuvînt de comandă canal 2
       OUT (12H),A
       LD  A,00H    ; Constanta de timp pentru canalul 2
       OUT (12H),A
       LD  A,0C7H   ; Cuvînt de comandă canal 3
       OUT (13H),A
       LD  A,01H    ; Constanta de timp pentru canalul 3
       OUT (13H),A
       ...
       EI           ; Validare întreruperi
       RETI         ; Revenire din subrutina de serviciu
```

Secvenţa MODT programează canalul 0 în modul temporizator şi canalele 1, 2, 3 în modul numărător. Întreruperile sînt validate numai pentru canalul 3. Ca şi în cazul secvenţei anterioare de programare, se va preciza întîi conţinutul registrului vector de întrerupere şi, apoi, pentru fiecare canal în parte, al registrului de comandă şi al registrului constantă de timp. Cuvîntul de comandă 2FH, transmis canalului 0, are următoarea semnificaţie:

D_7=0 invalidează întreruperile pe canalul 0;

D_6=0 programează canalul 0 să lucreze în modul temporizator;

D_5=1 precizează că factorul de prescalare este 256;

D_4=0 specifică frontul negativ al impulsului de la intrarea CLK/TRG_0 ca front activ de lansare a operaţiei de măsurare a timpului;

D_3=1 specifică faptul că lansarea operaţiei de măsurare a timpului se va face din exterior, cu ajutorul unui impuls la intrarea CLK/TRG_0;

D_2=1 înseamnă că urmează scrierea constantei de timp;

D_1=1 specifică faptul că pentru canalul 0 funcţionarea va începe sau va fi reluată după încărcarea constantei de timp.

Cuvîntul de comandă 47H, transmis canalelor 1 şi 2, are aceeaşi semnificaţie ca în cazul secvenţei anterioare MDNUM, singura deosebire constînd în aceea că nu validează întreruperile. Acest cuvînt de comandă programează, aşa cum am mai spus, canalele 1 şi 2 să lucreze în modul numărător, numărătoarele lor fiind decrementate la fiecare impuls generat la ieşirile ZC/TO_0, respectiv ZC/TO_1. Cuvîntul de comandă 0C7H este identic cu cel din secvenţa MDNUM el programînd canalul 3, de asemenea în modul numărător, dar cu

întreruperi validate. Acest din urmă canal va fi utilizat numai pentru generarea unei întreruperi în momentul încheierii operației de cronometrare. De aceea, canalul se încarcă cu constanta 01H, întreruperea activîndu-se la terminarea măsurării timpului, cînd transmiterea din exterior a unui impuls la intrarea CLK/TRG_3 va conduce la decrementarea numărătorului-decrementor și trecerea lui prin zero.

Subrutina SRVCR citește numărătorul-decrementor al canalului 2 în care se găsește numărul cronometrat de secunde, de fapt restul scăderii acestui număr din constanta 00H, calculînd apoi valoarea reală a perioadei de timp măsurate. În continuare se afișează timpul cronometrat și se reinițializează cronometrul, transmițînd aceleași cuvinte de comandă și constante de timp pentru cele patru canale ale CTC, ca și în secvența MODT din programul principal.

1.4. CIRCUITUL PENTRU COMANDA INTRĂRILOR/IEȘIRILOR SERIE SIO-Z80

1.4.1. PREZENTARE GENERALĂ. COMUNICAȚII DE DATE

Circuitul pentru comanda intrărilor/ieșirilor serie SIO-Z80 face parte din familia de circuite Z80 și este destinat, în principal, implementării aplicațiilor de comunicații de date, asigurînd interfațarea independentă a două canale. Circuitul realizează interfața serială binară între echipamente de prelucrare de date și echipamente de comunicații, modemuri, convertind informația paralel-serie și serie-paralel și permițînd implementarea practic a tuturor protocoalelor de comunicație uzuale, de tip asincron și sincron, orientate pe caracter sau pe bit. SIO-Z80 este un dispozitiv de I/E deosebit de flexibil, programabil software, ce înglobează funcțiile unor circuite tradiționale de tip UART, *Universal Asynchronous Receiver/Transmitter*, receptor/transmițător asincron universal, cum sînt TMS6011, S1883, de tip USART, *Universal Synchronous/Asynchronous Receiver/Transmitter*, receptor/transmițător asincron /sincron universal, ca, de exemplu, 8251, 8251A, 2651, și ale unor circuite de comandă pentru comunicații sincrone ca 2652 sau 8273. Menționăm posibilitățile de lucru ale circuitului cu întreruperi vectorizate sau nevectorizate, în mod DMA sau în modul de testare software, *polling*. De asemenea, SIO-Z80 poate fi folosit, cu restricțiile impuse în primul rînd de viteza de transfer, pentru conectarea oricăror echipamente periferice cu interfețe serie, de pildă unități de discuri flexibile, imprimante sau console.

Dintre caracteristicile principale ale circuitului enumerăm:

— poate comanda independent două canale *duplex** ;

— vitezele de transmisie $0 \div 500$ Kbiți/s, pentru frecvența ceasului de 2,5 MHz (SIO-Z80) sau $0 \div 800$ Kbiți/s, pentru ceas de 4 MHz (SIO-Z80A);

---

* Transmiterea datelor simultan în ambele sensuri definește un canal *duplex;* transmiterea datelor într-un sens sau în celălalt alternativ, nesimultan, definește un canal *semiduplex*, iar transmiterea numai într-un singur sens caracterizează un canal *simplex*.

— *buffer* de recepţie cu patru niveluri şi *buffer* de transmisie cu două niveluri;

— mod de transmisie asincron cu 5, 6, 7 sau 8 biţi/caracter, 1, 1 $^1/_2$ sau 2 biţi de STOP, cu sau fără paritate, rate de transfer de x1, x16, x32 sau x64 perioada ceasului, generare şi detecţie de caractere BREAK, detecţie de erori de paritate, de depăşire sau de cadrare;

— mod de transmisie sincron cu sincronizare pe caracter, internă sau externă, generarea unuia sau a două caractere de sincronizare, inserarea automată de caractere de sincronizare, generarea şi verificarea de tip CRC; permite implementarea unor protocoale cum sînt BISYNC, SDLC, HDLC asigurînd inserarea delimitatorului, inserarea şi ignorarea de biţi „0", manipularea resturilor din cîmpul de informaţie;

— posibilitatea de a lucra în întreruperi conectat într-un lanţ de priorităţi.

Înainte de a trece la descrierea propriu-zisă a circuitului SIO-Z80, vom prezenta, pe scurt, cîteva noţiuni generale referitoare la realizarea structurilor ce pot fi conectate la canale de comunicaţii, pentru a transmite date la distanţă. Pentru o tratare detaliată trimitem la [14÷25].

Termenul *comunicaţii de date* se referă la transmisia electronică a informaţiei sau datelor codificate între două sau mai multe puncte, numite staţii de date, aflate la distanţă una de alta. Un proces de comunicaţii de date necesită, în general, cel puţin cinci elemente: un transmiţător sau o sursă de informaţii; un mesaj; o interfaţă serială binară; un canal sau o legătură de comunicaţii; un receptor al informaţiei transmise [14]. În figura 1.43 se dă schema generală a unui astfel de proces. Ca transmiţătoare sau receptoare se pot utiliza diferite terminale, echipamente de introducere-extragere de date, sisteme de calcul cu posibilităţi de teleprelucrare, alte echipamente specializate, toate din categoria echipamentelor terminale pentru prelucrarea datelor — ETPD [15]. Aceste echipamente pot include şi interfaţa serială binară necesară, în principal, pentru implementarea funcţiilor de conversie paralel-serie, la transmiţător, şi serie-paralel la receptor. Conversiile au fost impuse, după cum se cunoaşte, de adoptarea metodei seriale de transmisie pe canalele de comunicaţii, în vederea reducerii costului, atît prin minimizarea numărului de canale, cît şi prin eliminarea problemelor de sincronizare legate de recepţia

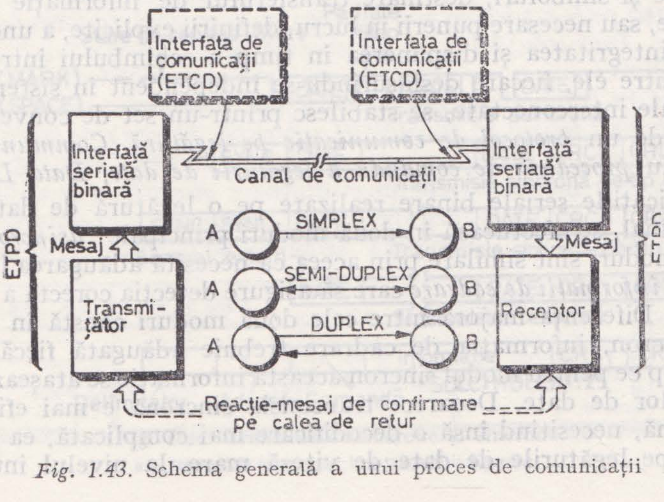


Fig. 1.43. Schema generală a unui proces de comunicaţii

datelor transmise paralel. La extremitățile canalului de comunicații este în general necesară conectarea unor echipamente specifice de adaptare cum sînt, de exemplu, adaptoarele telegrafice sau modemurile, numite global echipamente de terminare a circuitului de date — ETCD. Echipamentele ETCD realizează interfața de comunicații transformînd semnalele electrice binare cu care lucrează ETPD în semnale specifice canalului și invers. Canalul de comunicații poate fi definit ca o cale, un mediu, de transmisie a informației între două sau mai multe puncte terminale. În cazul transmisiei electronice, canalul de comunicații se materializează, de exemplu, printr-un conductor, un grup de conductori, un cablu coaxial sau o porțiune din spectrul de frecvențe radio. Un canal de comunicații împreună cu echipamentele terminale de tip ETCD formează un *circuit de date*. Adăugarea la acest circuit de date a dispozitivelor care realizează interfața serială binară conduce la implementarea unei *legături de date, Data Link*.

Pentru a se realiza transmiterea electronică de informații între două echipamente este mai întîi necesară *codificarea informației* sub forma unor combinații de simboluri binare. Prin această codificare, informației transmise (cifrele zecimale, literele alfabetului, semne aritmetice și de punctuație, diferite caractere de comandă) i se asociază în mod biunivoc o mulțime de configurații binare. Codul utilizat va reprezenta deci tocmai legea de corespondență între mulțimea simbolurilor vehiculate în echipamentele terminale și mulțimea cuvintelor binare asociate. Se cunosc mai multe coduri de acest fel, dintre care enumerăm: codul Baudot, cu cinci cifre binare, utilizat pentru transmisiile telegrafice de viteză mică, codurile BCD și EBCDIC, codul CCITT nr. 5, cu 7 biți, a cărui primă versiune a fost elaborată în 1963 sub denumirea ASCII și care este utilizat pentru transmisii de date prin rețeaua telefonică generală comutată, rețeaua telegrafică și rețelele publice de date, codurile КОИ-7 și КОИ-8, standardizate în țările CAER. Informația astfel codificată se poate transmite în blocuri, mai multe blocuri putînd constitui, la rîndul lor, un mesaj. Un *bloc* reprezintă o entitate din informația transmisă, tratată unitar din punctul de vedere al protecției împotriva erorilor și deci validată sau respinsă la intrare de unitatea de control a transmiterii. *Mesajul*, definit în strînsă dependență de aplicația de prelucrare de date, grupează un ansamblu de caractere și simboluri, destinate transferului de informație de la sursă la destinație, sau necesare punerii în lucru, definirii explicite, a unei prelucrări. Formatul, integritatea și dispunerea în timp a schimbului între procese ce comunică între ele, fiecare desfășurîndu-se independent în sisteme de calcul sau terminale interconectate, se stabilesc printr-un set de convenții și reguli specificate de un *protocol de comunicație pe legătură, Communication Link Protocol*, sau *procedură de comandă a legăturii de date, Data Link Control*.

Comunicațiile seriale binare realizate pe o legătură de date sau local, cu un terminal, se efectuează în două moduri principale: *asincron* și *sincron*. Cele două moduri sînt similare prin aceea că necesită adăugarea la informația utilă a unei *informații de cadrare* care să asigure detecția corectă a caracterelor la recepție. Diferența majoră între cele două moduri constă în aceea că, în modul asincron, informația de cadrare trebuie adăugată fiecărui caracter emis, în timp ce pentru modul sincron această informație se atașează blocurilor sau mesajelor de date. Deoarece transmisia sincronă e mai eficientă decît cea asincronă, necesitînd însă o decodificare mai complicată, ea se folosește, de obicei, pe legăturile de date de viteză mare, la nivelul interconectării

mașină-mașină, în timp ce modul asincron se utilizează pentru canalele de viteză mică, la nivelul relației om-mașină.

În sistemele care folosesc modul *asincron*, linia de transmisiuni se găsește în repaus în starea „1" logic sau MARK (vezi figura 1.44 *a*). Emisia unui caracter este precedată de trecerea liniei în starea „0" logic sau SPACE, pe durata unui bit, numit bit de START. Această trecere indică receptorului că urmează emisia unui caracter. Dispozitivul receptor detectează bitul de START și, în continuare, biții de date. După transmiterea caracterului, linia va fi trecută în starea MARK pe durata a 1, $1^1/_2$, sau 2 biți, numiți biți de STOP, pregătindu-se astfel pentru emisia unui nou caracter. Lungimea caracterelor transmise asincron variază în funcție de codul întrebuințat, de exemplu: cinci biți pentru codul Baudot, șapte pentru ASCII (opt prin adăugarea opțională a bitului de paritate) sau opt biți pentru codul EBCDIC. Procesul de emisie va fi astfel repetat *caracter cu caracter* pînă la terminarea transmiterii întregului mesaj, momentele de început ale fiecărui caracter succedîndu-se în *mod asincron*, la intervale diferite de timp. Precizăm faptul că aspectul asincron al transmisiei se referă la succesiunea emisiei caracterelor și nu la succesiunea biților în cadrul unui caracter, aceștia transmițîndu-se întotdeauna într-un mod periodic, sincron.

Transmisia *sincronă*, figura 1.44*b*, în loc să adauge biții de cadrare fiecărui caracter din mesaj, ca în cazul asincron, grupează caracterele în blocuri, cărora le alipește informația de cadrare. Această informație de cadrare este compusă din unul, două sau, uneori, mai multe caractere de sincronizare, SYNC, și servește, la recepție, pentru determinarea limitelor șirului de caractere. După transmiterea caracterului de sincronizare, procesul de comunicație sincronă continuă cu transmiterea caracter cu caracter a unui bloc de date, fără a mai fi necesară adăugarea biților de START-STOP, ca în modul asincron. Blocurile sînt compuse din caractere de sincronizare, un anumit număr de caractere de comandă sau de date, de obicei cuprins între 100 și 10 000, un caracter de terminare și unul sau două caractere pentru detecția erorilor, în figura 1.44*b* $CRC_{1,2}$. Pentru a asigura sincronizarea pe întreaga durată a transmiterii unui mesaj, se obișnuiește inserarea în șirul de date a unor caractere de sincronizare, la intervale de una sau două secunde. Între blocuri sau

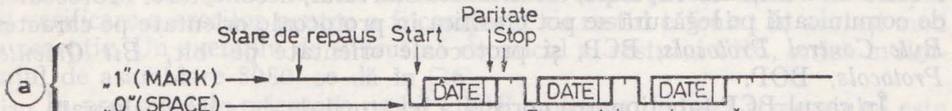


Fig. 1.44. Transmisii asincrone și sincrone:
a — transmisie asincronă; b — transmisii sincrone.

mesaje, linia de comunicații poate fi menținută în starea MARK sau se emit caracterele de sincronizare.

Diferențele între modurile de transmisie asincron și sincron conduc și la deosebiri între echipamentele ETCD corespunzătoare. Modemurile asincrone utilizează tehnici mai simple, cum este și cea a deplasării frecvenței prin comutare, *Frequency Shift Keying*, în fapt o modulație în frecvență, care implică, în cazul transmisiei, generarea unui ton audio, de o anumită frecvență, pentru MARK și a altuia, de altă frecvență, pentru SPACE. Modemul receptor detectează aceste tonuri pe linia telefonică, le convertește în semnale logice și le prezintă pe interfața ETCD-ETPD, către terminalul receptor. Deoarece funcționarea nu este condiționată de viteza de transmisie, modemul asincron poate lucra cu viteză variabilă de la zero pînă la viteza maximă. Spre deosebire de modul asincron, în cazul sincron se impune ca, la recepție, după detectarea caracterului de sincronizare, sincronizarea să fie menținută pe durata transmiterii întregului bloc de date. Aceasta necesită o tehnică particulară de formare în modem a unui semnal de sincronizare, o bază de timp, cu care să se eșantioneze informația recepționată. Semnalul de sincronizare emis pe interfața ETCD-ETPD se poate obține fie din datele recepționate, cu ajutorul unei bucle cu calare pe fază, *Phase Locked Loop* — PLL, fie dintr-o sursă externă. Existența semnalului de sincronizare face ca transferul datelor între modem și terminal să se desfășoare în sincronism cu această informație de timp. Datorită semnalului suplimentar de ceas, modemurile sincrone sînt concepute să lucreze numai la anumite rate prestabilite de transfer. În cazul utilizării unei bucle PLL, modemul receptor își calează frecvența pe aceea a transmițătorului, interpretînd schimbările de fază ca date.

Transmisia sincronă se utilizează pe legăturile de date de viteză mare. Datorită eficienței sale, este preferată transmisiei asincrone de majoritatea protocoalelor de comunicații pe legătură. Funcțiile importante ale acestor protocoale sînt: stabilirea și terminarea unei legături între două puncte; asigurarea integrității mesajelor prin detecția erorilor, cereri de retransmisie, confirmări pozitive sau negative; identificarea emițătorului și receptorului; funcții de comandă speciale cum sînt cereri de stare, inițializarea stației, confirmarea inițializării, start, confirmarea startului, deconectare. Protocoalele de comunicații pe legătură se pot clasifica în protocoale orientate pe caracter, *Byte Control Protocols*, BCP, și protocoale orientate pe bit, *Bit Oriented Protocols*, BOP.

În cazul BCP funcționarea corectă a legăturii de date este asigurată prin utilizarea unui *set definit de caractere de comandă*. Aceste caractere de comandă fac parte din codul utilizat, de exemplu ASCII sau EBCDIC. Mesajele de tip BCP se transmit în *blocuri* compuse dintr-un cîmp de comandă, antet sau *header*, un cîmp de text sau corp și un cîmp pentru verificarea erorilor sau terminator. Alcătuirea unui bloc BCP se prezintă în figura 1.45. Ca delimita-

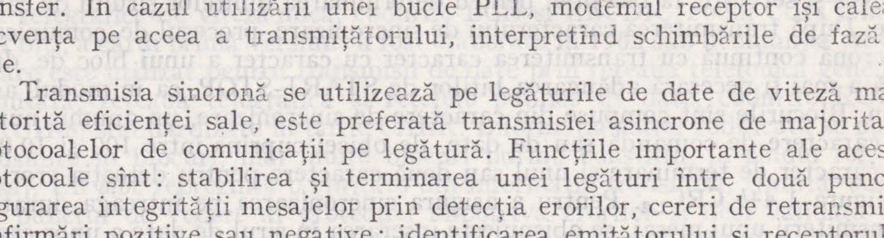


Fig. 1.45. Structura unui bloc BCP

toare de cîmp sau de bloc se utilizează caractere speciale cum sînt, de exemplu, pentru protocolul BISYNC, SOH — Început de antet, *Start of Header*, STX — Început de text, *Start of Text*, ETX — Sfîrșit de text, *End of Text*, EOT — Sfîrșit al transmiterii, *End of Transmission*. Pentru alte protocoale, cum este DDCMP, *Digital Data Communications Message Protocol*, dezvoltat de firma DEC, în vederea determinării cîmpului de text, se folosește tehnica numărării caracterelor.

Într-un bloc BCP, antetul conține informații auxiliare referitoare la adresa sursei sau destinației mesajului, numărul *job*-ului, dacă e cazul, tipul mesajului, comenzi sau date, acțiunea de comandă, o confirmare pozitivă sau negativă în vederea asigurării unei recepții neeronate a mesajului anterior. Acțiunile de comandă folosesc la inițializarea unei stații secundare, confirmarea unei recepții bune sau defectuoase a blocurilor, interogarea unei stații în cazul în care nu răspunde sau nu confirmă primirea unui mesaj într-o anumită perioadă de timp, la întreruperea unei secvențe de transfer. Comenzile se exprimă cu ajutorul unor caractere speciale sau al unor secvențe de caractere. Cîmpul de text, absent în mesajele de comandă, poate conține caractere din codul folosit sau caractere, secvențe, transparente față de acest cod. Aceasta se obține prin utilizarea unui mod de transmisie transparent, a cărui implementare e particulară protocolului. Necesitatea transparenței apare în situațiile cînd trebuie transmise date binare, BCD și în virgulă mobilă sau programe în cod-obiect. Asigurarea recepției corecte se face cu ajutorul cîmpului de verificare a erorilor. Fiecare bloc transmis este verificat la recepție într-un mod ce depinde de codul și funcțiile lui. Dintre metodele de verificare menționăm pe cea de tip vertical, VRC, *Vertical Redundancy Check*, în care se calculează paritatea fiecărui caracter, și pe cea de tip longitudinal, LRC, *Longitudinal Redundancy Check*, în care se calculează suma modulo 2 a tuturor caracterelor, aceasta transmițîndu-se în cîmpul de verificare a erorilor. Cele două tipuri de verificare se folosesc împreună, cu observația că anumite caractere de comandă și informația care se transmite în modul transparent nu intră în calculul LRC. Un alt tip de verificare, larg întrebuințată, este cea cunoscută sub numele CRC, *Cyclic Redundancy Check*, care presupune efectuarea împărțirii polinomiale a șirului de biți ce se transmite cu un anumit polinom specific protocolului. Restul acestei împărțiri se emite în cîmpul de verificare a erorilor pentru a fi comparat cu restul calculat în același mod la recepție. Un exemplu de subrutină de calcul al restului CRC, scrisă în limbajul de asamblare 8080, se dă în [26].

În protocoalele orientate pe bit, BOP, unitatea de bază a mesajului este *cadrul*. Structura unui astfel de cadru BOP se dă în figura 1.46. Protocoalele BOP folosesc numai *unul sau două caractere specifice de comandă* pentru a asigura funcționarea corectă a legăturii de date. De exemplu, delimitarea începutului și sfîrșitului de cadru se face cu ajutorul unui caracter unic numit delimitator, FLAG. După recepționarea unui astfel de delimitator, se utili-

Delimitator	Adresă	Comandă	Informație	Cîmp verificare erori	Delimitator
01111110	1 sau mai multe caractere de 8 biți	1 sau 2 caractere de 8 biți	0 sau mai multe caractere (oricîte pentru ADCCP/HDLC, multiplu de 8 biți pentru SDLC)	Rest împărțire polinom CRC-16/CCITT	01111110

Fig. 1.46. Structura unui cadru BOP

zează semnificația pozițională pentru a despărți în cîmpuri secvența de biți ce urmează. Aceste cîmpuri sînt, după cum se poate vedea în figura 1.46, cîmpul de adresă, cîmpul de comandă, cîmpul de informație și cîmpul de verificare a erorilor. Cîmpurile de adresă, comandă și verificare a erorilor sînt de lungime fixă, în timp ce cîmpul de informație este de lungime variabilă. Ca exemple de protocoale BOP enumerăm: SDLC, *Synchronous Data Link Control,* — IBM, ADCCP, *Advanced Data Communication Control Procedures,* —ANSI, HDLC, *High-Level Data Link Control,*—ISO, BDLC, *Burroughs'Data Link Control.*

Fiecare din stațiile în funcțiune, conectate la o legătură de date, caută în permanență secvența-delimitator și apoi cîmpul de adresă pentru validarea recepției în cazul recunoașterii adresei proprii. Cînd transmisia se face de către stația primară, de control, cîmpul de adresă desemnează care stație secundară, controlată, trebuie să recepționeze cadrul. Cînd transmite o stație secundară, cîmpul de adresă comunică stației primare ce stație secundară a emis cadrul. O stație secundară trebuie să-și recunoască adresa înainte de a accepta un cadru. De asemenea, stația primară va accepta un cadru numai dacă acesta are în cîmpul de adresă numărul unei stații căreia i se dăduse anterior permisiunea de a transmite. Pentru a se asigura integritatea datelor ce se transmit, cîmpul de adresă apare în fiecare cadru. Aceasta conduce și la o creștere a flexibilității deoarece stația primară poate recepționa cadre întrețesute de la mai multe stații secundare. Cîmpul de comandă determină tipul mesajului, numărul de secvență al cadrului în emisie și recepție, o comandă de interogare de la stația primară sau un răspuns de la stația secundară. Stația primară utilizează cîmpul de comandă pentru a indica stației secundare adresate ce operație urmează să realizeze, în timp ce o stație secundară îl folosește pentru a răspunde stației primare. Cîmpul de informație are o lungime variabilă, datele putînd fi transmise în orice cod. Deoarece nu există restricții în ceea ce privește succesiunea de biți ce pot fi transmiși între delimitatoarele de început și sfîrșit ale unui cadru apare posibilă emisia a șase biți „1" succesivi, ceea ce s-ar interpreta la recepție ca un nou delimitator și ar conduce la terminarea incorectă a unui cadru. Pentru a elimina această situație, după emisia delimitatorului de început, stația transmițătoare numără biții „1" succesivi și, în cazul apariției a cinci biți „1" succesivi, inserează în mod automat un bit „0". La recepție, dacă apar cinci biți „1" consecutivi se așteaptă al șaselea și, dacă acesta este „0", se elimină, continuîndu-se recepția; dacă este „1", se consideră apariția delimitatorului de sfîrșit de cadru. Această transparență a conținutului unui cadru în raport cu codul folosit este una din caracteristicile importante ale protocoalelor de tip BOP, ea permițînd utilizatorului adoptarea oricărui format sau cod în funcție de aplicație. În transmisiile BOP nu caracterele au un înțeles specific ci biții. Cîmpul de verificare a erorilor rezultă din aplicarea unei metode de verificare de tip CRC și conține restul împărțirii polinomului construit cu ajutorul șirului de date transmise, la un polinom generator. În calculul restului intră toate datele transmise între două delimitatoare, cu excepția biților „0" inserați.

Un ultim aspect important, legat de problema implementării unui proces de comunicații de date, pe care îl vom prezenta pe scurt în această introducere, este cel al interfeței electrice între echipamentele ETPD și ETCD. Odată serializată și împachetată de către ETPD, conform modului și protocolului alese, informația trebuie să fie transferată modemului pentru a fi emisă pe

linie. Analog se procedează și în cazul recepției. Legătura cu modemul, specificată în diverse avize sau standarde ale Comitetului Consultativ Internațional de Telegrafie și Telefonie — CCITT, ale ISO sau ale EIA, *Electronic Industries Association*, impune existența pe interfață a unor semnale grupate în semnale de date, comenzi și sincronizare. Cea mai întrebuințată interfață între ETPD și ETCD este cea cunoscută sub denumirea EIA-RS-232C, acoperită și de avizul V.34 al CCITT. Acest aviz standardizează o interfață cu posibilități mai mari față de norma EIA-RS-232C stabilind atît circuitele de utilizare generală cuprinse în seria 100 cît și pe cele destinate apelului automat, prevăzute în seria 200. Interfața se utilizează pentru viteze de transmisie ce nu depășesc 20 Kbiți pe secundă. Conectarea se face cu conectoare de 25 de contacte, tip RK 25 fabricate de întreprinderea CONECT, prin intermediul unui cablu ce nu va depăși 15 metri. Conectorul „tată" se află pe echipamentul de tip ETPD, cu care se furnizează de obicei și cablul, iar conectorul „mamă" se montează pe echipamentul de tip ETCD, pe modem. Componența unui conector pentru interfața EIA-RS-232C este dată în tabelul 1.13. După cum se observă semnalele acestei interfețe sînt grupate în patru categorii: masă (AX), date (BX), comenzi (CX) și sincronizări (DX). Semnalele de date și comenzi pot fi semnale principale (BX, CX) sau secundare (SBX, SCX). Vom da în continuare cîteva explicații pentru fiecare din semnalele interfeței *EIA-RS-232C*.

TABELUL 1.13. **Componența unui conector EIA-RS-232C**

Numărul pinului din conector	Numărul circuitului CCITT	Circuitul echivalent EIA	Sensul de emisie ETPD ETCD	Denumirea semnalului
1	101	AA		Masă de protecție
7	102	AB		Masă de semnalizare sau retur comun
2	103	BA	→	Emisie date
3	104	BB	←	Recepție date
4	105	CA	→	Cerere pentru emisie
5	106	CB	←	Gata de emisie
6	107	CC	←	Modem gata
20	108.2	CD	→	Terminal de date gata
22	125	CE	←	Detector de apel
8	109	CF	←	Detector purtătoare
21	108.1	CG	→	Conectează modem la linie
	(110)		(←)	(Detector de calitate a semnalului)
23	111	CH	→	Selector al vitezei de transmisie (ETPD)
23	112	CI	←	Selector al vitezei de transmisie (ETCD)
24	113	DA	→	Bază de timp pentru emisie ETPD
15	114	DB	←	Bază de timp pentru emisie ETCD
17	115	DD	←	Bază de timp pentru recepție ETCD
14	118	SBA	→	Emisie date pe calea de retur
16	119	SBB	←	Recepție date pe calea de retur
19	120	SCA	→	Cerere pentru emisie pe calea de retur
13	121	SCB	←	Calea de retur gata
12	122	SCF	←	Detector purtătoare pe calea de retur

Note: Pinii 9 și 10 sînt rezervați pentru testarea modemului.
Pinii 11, 18 și 25 nu sînt utilizați.
Semnalul afectat pinului 21 depinde de construcția modemului.

Masă de protecție. Conductorul se leagă la șasiul aparatului putînd fi eventual conectat la împămîntarea generală.

Masă de semnalizare sau retur comun. Conductorul stabilește potențialul de referință, 0V, pentru ambele aparate.

Emisie date. Pe acest circuit se generează semnale de către ETPD pentru a fi transmise la distanță.

Recepție date. Semnalele generate pe acest circuit provin de la modemul care, la rîndul lui, le-a recepționat pe linia de comunicații.

Cerere pentru emisie. Semnalul generat pe acest circuit se utilizează pentru a comanda modemul local să lucreze în transmisie și, pe un canal semi-duplex, pentru a controla direcția transmisiei de date din echipamentul ETCD local.

Gata de emisie. Semnalul se generează de către modem pentru a indica dacă acesta este pregătit să transmită date.

Modem gata. Indică starea modemului local, *on-line* sau *off-line.*

Terminal de date gata. Semnalul generat către ETCD este utilizat pentru conectarea echipamentului ETCD la canalul de comunicație. Activarea acestui semnal pregătește conectarea modemului și menține conectarea stabilită prin mijloace externe ca, de exemplu, apelare manuală, răspuns manual sau apel automat.

Detector de apel. Activarea acestui circuit indică recepția pe canalul de comunicație a unui semnal de apel.

Detector purtătoare. Activarea circuitului se face atunci cînd modemul recepționează un semnal care îndeplinește anumite criterii stabilite de fabricant. Dezactivarea circuitului înseamnă fie că modemul nu a recepționat nici un semnal, fie că semnalul recepționat nu respectă criteriile stabilite, neputînd fi deci demodulat.

Detector de calitate a semnalului. Semnalul generat pe acest circuit se folosește pentru a indica existența foarte probabilă a unei erori în șirul de date recepționate. Dezactivarea circuitului indică o mare probabilitate de eroare. Semnalul poate fi întrebuințat în unele situații pentru a comanda retransmisia semnalului precedent de date.

Conectează modem la linie. Semnalul se utilizează pentru a comanda modemului conectarea la linie.

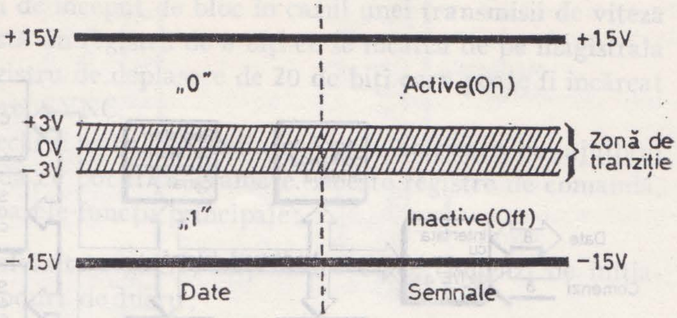
Selector al vitezei de transmisie (ETPD). Semnalul este utilizat pentru a selecta una din cele două viteze de transmisie în cazul modemurilor sincrone care lucrează cu două viteze, sau una din cele două game de viteze, în cazul modemurilor asincrone cu două viteze. Semnalul activ selectează viteza sau gama de viteze mai mare.

Selector al vitezei de transmisie (ETCD). De asemenea, utilizat pentru a selecta între două viteze de transmisie, în cazul modemurilor sincrone, sau între două game de viteze pentru modemurile asincrone. Semnalul activ selectează viteza sau gama de viteze mai mare.

Bază de timp pentru emisie ETPD. Semnalele generate de ETPD pe acest circuit asigură modemul la emisie cu informația de timp necesară. Tranziția din starea activă în starea inactivă indică în mod nominal centrul fiecărui bit de date.

Bază de timp pentru emisie ETCD. Semnalele generate de ETCD pe acest circuit asigură informația de timp, de sincronizare, pentru terminalul

Fig. 1.47. Nivelurile de tensiune ale semnalelor de pe interfaţa EIA-RS-232C



de date, ETPD. ETPD va trebui să asigure generarea unui semnal de date la care tranziţiile între biţi vor apărea în mod nominal în momentul tranziţiilor din starea inactivă în starea activă ale semnalului *Bază de timp pentru emisie ETCD.*

Bază de timp pentru recepţie ETCD. Semnalele generate de ETCD pe acest circuit reprezintă informaţia de timp care însoţeşte datele recepţionate către ETPD. Tranziţia circuitului din starea activă în starea inactivă indică în mod nominal centrul fiecărui bit de date recepţionat.

Funcţiile semnalelor secundare sînt aceleaşi cu cele ale semnalelor principale. Semnalele de sincronizare, grupa DX, sînt întrebuinţate numai în transmisiile sincrone. Legăturile electrice pe interfaţa EIA-RS-232C (CCITT V.24) sînt efectuate în tensiune, nivelurile fiind cele din figura 1.47. Fronturile nu trebuie să depăşească 30 V/μs, iar timpul de trecere prin zona de tranziţie, în care semnalul este nedefinit, nu trebuie să fie mai mare de 1 ms sau de 4% din durata unui element de semnal. Pentru conectarea echipamentelor realizate cu interfaţă EIA-RS-232C, de exemplu modemuri, cu echipamente care lucrează în niveluri TTL, sînt necesare circuite de adaptare, cum sînt şi ROB 1488, ROB 1489 fabricate de ICCE. Mai menţionăm că interfaţa EIA-RS-232C se utilizează, ca interfaţă serie standard, şi în alte aplicaţii decît cele din domeniul comunicaţiilor, de exemplu în conectarea la sisteme de calcul a unor echipamente periferice uzuale, cum sînt consolele sau imprimantele.

1.4.2. STRUCTURA CIRCUITULUI SIO-Z80

Schema bloc a circuitului SIO-Z80 se prezintă în figura 1.48. Aşa cum se observă, circuitul este organizat pe principiul unei magistrale interne, la care sînt conectate blocurile funcţionale: interfaţa cu magistrala de I/E a unităţii centrale, logica de comandă a întreruperilor, logica de comandă internă, canalele seriale *duplex* A şi B. Fiecare din cele două canale este constituit, la rîndul lui, din registre de stări şi comenzi, logica de stări şi comenzi pentru interfaţa cu modemul sau cu alte echipamente externe, partea de transmisie/recepţie propriu-zisă. Alcătuirea acestei părţi de transmisie/recepţie pentru canalul A se dă în figura 1.49. La recepţie se observă cele patru niveluri ale *buffer*-ului, trei alcătuind o stivă de tip FIFO, *First-InFirst-Out,* iar al patrulea fiind reprezentat de registrul pentru conversia serie-paralel.

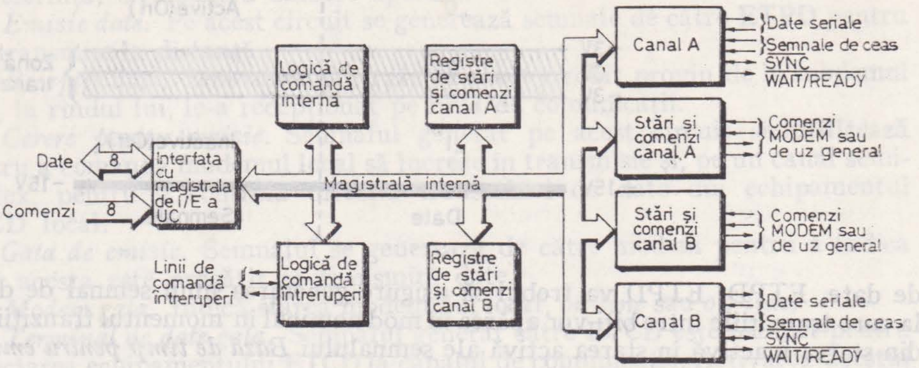


Fig. 1.48. Schema bloc a circuitului SIO-Z80

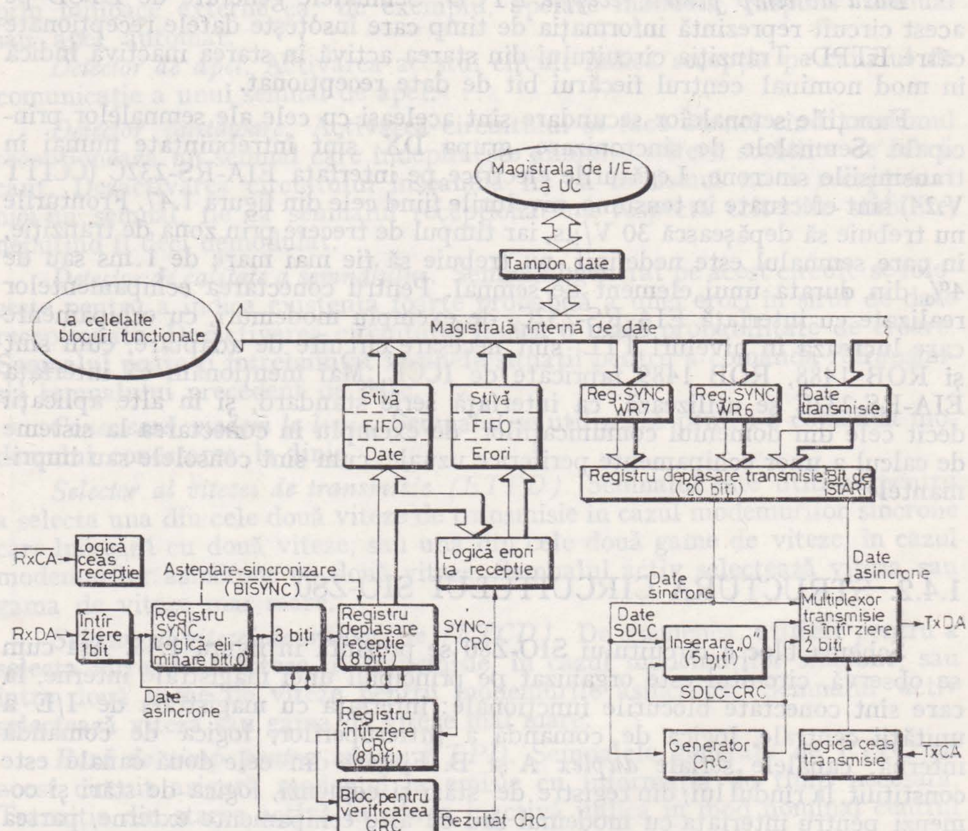


Fig. 1.49. Organizarea canalului serial A

Această memorie tampon permite o întîrziere, necesară, de exemplu, pentru tratarea unei întreruperi de început de bloc în cazul unei transmisii de viteză mare. La transmisie există un registru de 8 biți ce se încarcă de pe magistrala internă de date și un registru de deplasare de 20 de biți care poate fi încărcat cu date sau cu caractere SYNC.

Funcționalitatea fiecărui canal se definește prin software cu ajutorul a opt registre de comandă ce pot fi programate. Aceste registre de comandă, WR0÷WR7, au următoarele funcții principale:

WR0 — adresare celelalte registre, inițializare CRC, comenzi de inițializare pentru diverse moduri de lucru;

WR1 — definire mod de transfer și întreruperi transmisie/recepție;

WR2 — vector de întrerupere (scris numai pentru canalul B);

WR3 — comenzi și parametri recepție;

WR4 — moduri și parametri diverși transmisie/recepție;

WR5 — comenzi și parametri transmisie;

WR6 — caracter SYNC sau cîmp de adresă SDLC;

WR7 — caracter SYNC sau delimitator SDLC.

Pentru a cunoaște starea fiecărui canal, programatorul poate citi trei registre de stare. Aceste trei registre, RR0÷RR2, dau informații despre condițiile de eroare, vectorul de întrerupere sau starea unor semnale de interfață standard:

RR0 — stare *buffer* de transmisie/recepție, stare de întrerupere, stare externă;

RR1 — stare condiție de recepție specială;

RR2 — vector de întrerupere modificat (citit numai pe canalul B)

Observăm că vectorul de întrerupere al circuitului se scrie numai în registrul WR2 al canalului B și se citește, modificat de starea dispozitivului, în registrul RR2, de asemenea al canalului B.

1.4.3. DESCRIEREA CONEXIUNILOR EXTERNE

Datorită constrîngerilor impuse de necesitatea de a împacheta circuitul într-o capsulă cu 40 de conexiuni externe SIO-Z80 se fabrică în trei versiuni de încapsulare. Conexiunile externe ale variantei preferate în majoritatea aplicațiilor, SIO-Z80/2, sînt date în figura 1.50. În tabelul 1.14 sînt date configurațiile conexiunilor care diferă pentru cele trei variante: SIO-Z80/2 nu are $\overline{\text{SYNCB}}$, lui SIO-Z80/1 îi lipsește semnalul $\overline{\text{DTRB}}$, iar la SIO-Z80/0 semnalele $\overline{\text{TxCB}}$ și $\overline{\text{RxCB}}$ sînt legate împreună.

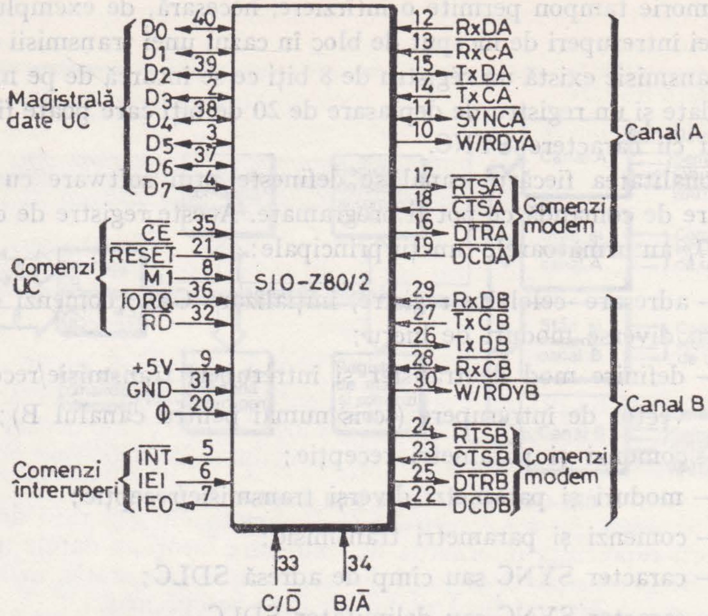


Fig. 1.50. Conexiunile externe ale circuitului SIO-Z80/2

Vom da în continuare semnificația semnalelor specifice circuitului, diferite de acelea care permit conectarea lui la sisteme realizate cu microprocesorul Z80. Aceste din urmă semnale, $D_0 \div D_7$, $\overline{CE}$, $\overline{M1}$, $\overline{IORQ}$, $\overline{RD}$, Φ, $\overline{INT}$, IEI și IEO, sînt identice cu cele descrise în § 1.2 și 1.3.

$B/\overline{A}$, selecție canal A sau B. Intrare cu ajutorul căreia se precizează canalul cu care UC face transferul pe timpul unei operații de I/E. De obicei, la această conexiune se leagă bitul A_0 al magistralei de adrese. „1" selectează canalul B, iar „0" canalul A.

$C/\overline{D}$, selecție date sau comenzi/stări. Intrare prin intermediul căreia se determină tipul informației transferate între UC și SIO pe timpul unei operații de I/E. $C/\overline{D}=1$ pe durata unei scrieri I/E va conduce la interpretarea de către SIO a informației de pe magistrala de date $D_0 \div D_7$ ca o comandă, iar $C/\overline{D}=0$

TABELUL 1.14. Conexiuni externe diferite ale variantelor SIO-Z80

Conexiune externă	SIO-Z80/2	SIO-Z80/1	SIO-Z80/0
25	$\overline{DTRB}$	TxDB	$\overline{DTRB}$
26	TxDB	TxCB	TxDB
27	TxCB	RxCB	RxTxCB
28	RxCB	RxDB	RxDB
29	RxDB	$\overline{SYNCB}$	$\overline{SYNCB}$

va face ca informaţia să fie considerată date. O citire cu $C/\overline{D}=1$ înseamnă o citire de stare. De obicei, la această conexiune se leagă bitul A_1 al magistralei de adrese.

$\overline{RESET}$, iniţializare. Intrare activă pe „0". La activarea acestei linii se invalidează circuitele de recepţie şi de transmisie, se forţează ieşirile de emisie date TxDA şi TxDB în starea MARK, se trec pe „1" toate comenzile către modem şi se invalidează întreruperile. După o astfel de iniţializare, înainte de a recepţiona sau transmite date, este necesară reprogramarea tuturor registrelor de comenzi.

RxDA, RxDB, *Receive Data*, recepţie date. Intrări serie active pe „1", ce admit semnale cu niveluri TTL.

$\overline{RxCA}$, $\overline{RxCB}$, *Receiver Clock*, baze de timp pentru recepţie. Datele recepţionate se eşantionează cu frontul pozitiv al semnalelor primite la aceste intrări. Ceasurile $\overline{RxC}$ pot avea, în modurile asincrone, o frecvenţă egală, de 16, 32 sau 64 ori mai mare decît viteza de transmisie a datelor. SIO-Z80 nu are nevoie de semnale de ceas cu factor de umplere egal, ceea ce permite generarea lor cu ajutorul unui circuit CTC-Z80. Cele două intrări sînt de tip *trigger-Schmitt*, fără o margine garantată a nivelului de zgomot, admiţînd fronturi pozitive şi negative lente.

TxDA, TxDB, *Transmit Data*, emisie date. Ieşiri serie, niveluri TTL, active pe „1". Datele generate la aceste ieşiri sînt schimbate pe fronturile negative ale ceasurilor $\overline{TxC}$.

$\overline{TxCA}$, $\overline{TxCB}$, *Transmitter Clocks*, baze de timp pentru emisie. Intrări de ceas de tip *trigger-Schmitt* care, ca şi ceasurile de recepţie, pot avea în modurile asincrone frecvenţa egală, de 16, 32 sau 64 ori mai mare decît rata de transfer. Este obligatoriu ca factorul de multiplicare pentru emiţător şi receptor să fie acelaşi. De asemenea, $\overline{TxC}$ se poate obţine cu ajutorul unui CTC.

$\overline{SYNCA}$, $\overline{SYNCB}$, *Synchronization*, sincronizare. Intrări/ieşiri active pe „0". În timpul recepţiei asincrone aceste conexiuni externe sînt folosite ca intrări similare cu $\overline{CTS}$ şi $\overline{DCD}$. În acest mod, afectează biţii de stare SYNC/aşteptare-sincronizare din registrele de stare RR0, fiind utilizabile pentru orice funcţie de intrare. În plus, SIO sesizează tranziţiile de nivel şi întrerupe UC Precizăm că, dacă se utilizează modul asincron şi întreruperile (externe/de stare) sînt validate, intrarea SYNC nu trebuie lăsată în gol pentru a nu se genera întreruperi parazite. În modul de sincronizare externă, $\overline{SYNCA}$, $\overline{SYNCB}$ sînt tot intrări comandate pe „0" cu al doilea front pozitiv al semnalului $\overline{RxC}$ după frontul pozitiv al $\overline{RxC}$ cu care s-a eşantionat ultimul bit al caracterului de sincronizare. Cu alte cuvinte, după detecţia configuraţiei de sincronizare, logica externă trebuie să aştepte două perioade complete ale ceasului de recepţie şi apoi să activeze intrările SYNC. După trecerea pe „0" intrarea SYNC trebuie menţinută în această stare pînă cînd UC informează logica externă de detecţie a sincronizării că s-a pierdut sincronizarea sau că poate să înceapă un nou mesaj. În modul de sincronizare externă, asamblarea caracterului începe cu primul front pozitiv al $\overline{RxC}$ după activarea intrării SYNC. În modul de sincronizare internă, MONOSYNC sau BYSYNC, cele

două conexiuni externe sînt utilizate ca ieșiri activate pe durata unei părți din perioada ceasului, $\overline{\text{RxC}}$, în care sînt recunoscute caracterele de sincronizare. Recunoașterea caracterelor de sincronizare nu se memorează, astfel încît ieșirile $\overline{\text{SYNC}}$ vor fi activate de fiecare dată cînd se recunoaște o configurație de sincronizare.

$\overline{\text{W/RDYA}}$, $\overline{\text{W/RDYB}}$, *Wait/Ready A*, *Wait/Ready B*, așteptare/gata A,B. Ieșiri de tip drenă-în-gol, atunci cînd sînt programate pentru a realiza funcția $\overline{\text{WAIT}}$, sau comandate pe „1" și „0", cînd sînt programate pentru funcția $\overline{\text{READY}}$. Se programează ca ieșiri de tip $\overline{\text{READY}}$, pentru a comanda circuite DMA, de exemplu 8257, și ca ieșiri de tip $\overline{\text{WAIT}}$, pentru a sincroniza unitatea centrală la rata de transmisie a SIO. La inițializare ieșirile se trec în starea drenă-în-gol, $\overline{\text{WAIT}}$.

$\overline{\text{RTSA}}$, $\overline{\text{RTSB}}$, *Request To Send*, cerere pentru emisie. Ieșiri active pe „0". Ieșirea trece pe „0" atunci cînd bitul RTS din registrul de comandă WR5 este programat pe „1". Dacă se utilizează modul de transmisie asincron și bitul RTS din WR5 este programat pe „0", ieșirea va trece pe „1" la golirea *buffer*-ului de transmisie. În modurile sincrone, conexiunea $\overline{\text{RTS}}$ urmărește starea bitului RTS din registrul WR5. Cele două ieșiri se pot întrebuința și ca ieșiri de comandă de uz general.

$\overline{\text{CTSA}}$, $\overline{\text{CTSB}}$, *Clear To Send*, gata de emisie. Intrări active pe „0". Dacă bitul D_5, auto-validare, din registrul WR3, se programează pe „1", trecerile acestor linii pe „0" validează circuitele de transmisie respective. Programarea pe „0" a bitului D_5 permite folosirea lor ca intrări de uz general, ce pot fi citite în RR0. Ambele conexiuni sînt de tip *trigger-Schmitt*, permițînd comandarea lor cu semnale avînd fronturi lente. SIO poate detecta tranzițiile pozitive sau negative ale semnalelor aplicate la intrările CTS și apoi, dacă a fost programat corespunzător, să genereze întreruperi către UC.

$\overline{\text{DTRA}}$, $\overline{\text{DTRB}}$, *Data Terminal Ready*, terminal de date gata. Ieșiri active pe „0" care urmăresc starea programată a bitului DTR din registrul WR5. De asemenea, ele pot fi folosite ca ieșiri de comandă de uz general.

$\overline{\text{DCDA}}$, $\overline{\text{DCDB}}$, *Data Carrier Detect*, detector purtătoare. Intrări active pe „0" de tip *trigger-Schmitt*, care, dacă se lucrează în modul auto-validare, bitul D_5 din WR3 programat pe „1", servesc la validarea recepției. Pot fi utilizate și ca intrări de uz general, ținînd seama și de faptul că SIO detectează tranzițiile de nivel, generînd, dacă a fost programat, întreruperi către UC.

1.4.4. FUNCȚIONAREA CIRCUITULUI. POSIBILITĂȚI DE LUCRU

SIO-Z80 asigură comanda a două canale *duplex* independente, în oricare din procedurile actual întîlnite în comunicațiile de date sincrone sau asincrone. Funcționarea circuitului este determinată de conținutul registrelor de comandă, ce trebuie programate înainte ca SIO să execute vreo funcție. Registrele de stare se pot investiga în orice moment, dar numai unele comenzi și moduri pot fi modificate pe durata unei funcționări deja programate.

1.4.4.1. MODURI DE LUCRU CU UNITATEA CENTRALĂ

Pentru realizarea transferului de date, stări şi comenzi de la şi spre UC, SIO-Z80 foloseşte testarea software, *polling*, întreruperile sau tehnica transferului pe blocuri de octeţi. Transferul pe blocuri de octeţi se poate efectua prin intermediul unităţii centrale, cu ajutorul instrucţiunilor I/E de transfer pe bloc, de exemplu OTIR, sau prin intermediul unor circuite specializate de tip DMA.

Tehnica testării software, polling, impune verificarea permanentă prin program a unor parametri specifici, ce caracterizează transferul. SIO-Z80 are asociate fiecărui canal două registre separate de stare, RR0, RR1, şi un registru comun cu vectorul de întrerupere modificat, RR2, accesibil numai pe canalul B. Cele două registre de stare sînt actualizate de logica internă a circuitului în momente definite în raport cu funcţiile realizate. Unul din registrele de stare se foloseşte pentru a indica unităţii centrale momentele cînd SIO are pregătite date sau cînd are nevoie de date. De asemenea, acelaşi registru memorează şi condiţiile de eroare. Al doilea registru de stare, ce păstrează condiţiile de recepţie speciale, nu trebuie citit decît după ce s-a recepţionat un caracter. În modul *polling* toate întreruperile circuitului se invalidează.

Lucrul în întreruperi cu SIO-Z80 este facilitat de modul elaborat în care circuitul poate genera întreruperi, permiţînd utilizarea lui eficientă în aplicaţii de timp real. SIO are un registru, WR2, în care UC înscrie vectorul de întrerupere şi un altul RR2, ce păstrează vectorul modificat. Ambele registre sînt accesibile numai pe canalul B. Dacă circuitul se programează corespunzător, schimbarea de stare conduce la modificarea unei zone de trei biţi din vectorul de întrerupere astfel încît, acesta poate „puncta" direct opt rutine de tratare diferite, eliminîndu-se timpul consumat cu operaţiile de analiză a stării circuitului.

Întreruperile generate de SIO-Z80 se împart în trei grupe, de recepţie, de transmisie şi externe/de stare, în această ordine de prioritate pe fiecare canal, canalul A fiind prioritar faţă de canalul B. Fiecare sursă de generare a întreruperii poate fi validată prin program. Dacă se validează *întreruperile de transmisie,* UC va fi întreruptă la golirea *buffer*-ului de transmisie, după ce, bineînţeles, el a fost încărcat cu un caracter. *Întreruperile de recepţie* pot fi de două feluri: *întreruperi-la-primul-caracter-recepţionat* şi *întreruperi-la-fiecare-caracter-recepţionat.* Primul fel de întreruperi de recepţie se utilizează de obicei în cazul modului de transfer pe blocuri de octeţi. Întreruperea-la-fiecare-caracter-recepţionat permite modificarea vectorului în situaţia detectării unei erori de paritate. Ambele tipuri de întrerupere admit generarea întreruperilor datorită *condiţiilor-speciale-de-recepţie**, spre exemplu, întrerupere de sfîrşit-de-cadru în procedura SDLC. O condiţie de recepţie specială poate conduce la generarea unei întreruperi, numai dacă în prealabil a fost selectat unul din cele două tipuri de întrerupere, la primul caracter recepţionat sau la fiecare caracter recepţionat. În modul întrerupere-la-primul-caracter-recepţionat, condiţiile speciale de recepţie, cu excepţia erorii de paritate, generează întreruperi şi după activarea întreruperii la recepţia primului caracter, de pildă întrerupere de depăşire la recepţie. *Întreruperile externe/de stare* au

---

* Vezi şi § 1.4.5.1.

funcţia de a supraveghea tranziţiile semnalelor de la intrările $\overline{\mathrm{CTS}}$, $\overline{\mathrm{DCD}}$ şi $\overline{\mathrm{SYNC}}$. O întrerupere externă/de stare poate fi cauzată de necesitatea de emisie a CRC sau de detecţia, în şirul de date recepţionate, a unei secvenţe BREAK, în comunicaţiile asincrone, ori a unei secvenţe ABORT, în protocolul SDLC. Întreruperea declanşată de o secvenţă BREAK/ABORT permite circuitului SIO să atenţioneze unitatea centrală la detecţia sau la terminarea unei asemenea secvenţe. Astfel, întreruperile externe/de stare înlesnesc terminarea corectă a mesajului curent, iniţializarea următorului mesaj precum şi detectarea oportună a secvenţelor BREAK/ABORT.

Dacă unitatea centrală a sistemului e realizată cu microprocesorul Z80, lucrul în întreruperi se desfăşoară conform modului 2, SIO generînd vectorul în ciclul de achitare, vezi § 1.1, pentru a forma, împreună cu conţinutul registrului I, adresa unei locaţii unde se află adresa de început a subrutinei de serviciu.

Transferul pe blocuri de octeţi poate fi implementat fie cu ajutorul unităţii centrale, prin utilizarea instrucţiunilor de I/E pe bloc, de exemplu OTIR, OTDR, INIR sau INDR pentru Z80, fie prin folosirea unor circuite specializate pentru acces direct la memorie, DMA, ca DMA-Z80 sau 8257. Acest mod de lucru întrebuinţează ieşirea $\overline{\mathrm{W/RDY}}$, comandată cu ajutorul a trei biţi programabili, D_7, D_6, D_5, din registrul de comandă WR1. Ieşirea poate îndeplini funcţia $\overline{\mathrm{WAIT}}$ pentru ca, prin conectarea la intrarea corespunzătoare a microprocesorului, să permită transferul pe bloc cu ajutorul instrucţiunilor specializate de I/E, prelungind ciclii de I/E atunci cînd SIO nu e pregătită de transfer. Cînd se utilizează circuite DMA, ieşirea realizează funcţia $\overline{\mathrm{READY}}$ indicînd circuitului specializat că SIO este gata pentru un transfer cu memorie.

1.4.4.2. MODURI DE TRANSMISIE ASINCRONE

Transmisia şi recepţia se realizează independent pe fiecare din cele două canale. Caracterele pot avea $5 \div 8$ biţi şi, opţional, un bit de paritate. Partea de transmisie a circuitului SIO-Z80 poate adăuga fiecărui caracter 1, 1 $^1/_2$ sau 2 biţi de STOP şi poate genera in orice moment caracterul BREAK. SIO nu impune folosirea unor semnale bază de timp pentru transmisie şi recepţie cu factor de umplere 1/2 ceea ce permite utilizarea unor circuite cum e şi CTC-Z80. Rata de transmisie a datelor se poate selecta pentru a fi 1/1, 1/16, 1/32 sau 1/64 din frecvenţa ceasului. Conexiunea externă $\overline{\mathrm{SYNC}}$ este posibil de a fi programată ca intrare la care să se aplice un semnal de tipul *Detector de apel* generat de modem.

Transmisia începe numai după programarea pe „1" a bitului validare-transmisie, D_3, din registrul de comandă WR5*. Dacă a fost programat modul auto-validare, bitul D_5 din WR3, începerea transmisiei va fi condiţionată şi de activarea intrării $\overline{\mathrm{CTS}}$. Starea de repaus, în care SIO nu emite, este starea MARK, „1", pînă cînd se iniţiază o transmisie. La programarea în WR5 a bitului D_4, transmisie-BREAK, ieşirea TxD este forţată în starea SPACE, „0", indiferent de starea în care se găsea înaintea acestei comenzi.

* vezi § 1.4.5.1.

pentru a elibera linia, fie se anulează comanda transmisie-BREAK, fie se inițializează circuitul. Mai precizăm că, dacă se utilizează un cod cu 5 biți/caracter, biții nefolosiți ai fiecărui octet scris în SIO-Z80, D_5, D_6 și D_7, vor fi „0".

Recepția asincronă începe numai după ce bitul validare-recepție, D_0, din registrul de comandă WR3, a fost programat pe „1". Dacă se folosește modul auto-validare, e necesar, pentru a valida recepția, ca intrarea $\overline{DCD}$ să fie și ea activată. Recepția este protejată la tranzițiile pe „0" parazite, de scurtă durată, cu ajutorul unui mecanism de rejecție care testează semnalul după o jumătate de bit de la detecția nivelului „0" pe liniile de recepție, RxDA sau RxDB. Dacă nivelul „0" nu se menține, tranziția e parazită și procesul de asamblare a caracterului nu va mai fi declanșat. Această metodă de detecție a unui bit de START micșorează numărul erorilor în recepțiile pe linii zgomotoase.

Circuitele de recepție au, după cum se poate vedea și în figura 1.49, o stivă cu patru niveluri, incluzînd și registrul de deplasare, pentru stocarea datelor primite. Această stocare permite tratarea de către UC a unei întreruperi de început de bloc, în cazul unui transfer de viteză mare. Există, de asemenea, o stivă pentru erori, paralelă cu stiva de date, indicatorii de eroare fiind încărcați odată cu caracterele la care se referă. Indicatorii pot fi citiți prin program aflîndu-se în registrul cu condițiile speciale de recepție, RR1*. Indicatorii eroare-depășire-recepție și eroare-paritate sînt cumulativi și nu pot fi șterși decît prin comanda 6, inițializare-erori **. Pe de altă parte, indicatorii sfîrșit-de-cadru și eroare-de-CRC/cadrare reflectă starea curentă a caracterului aflat în *buffer* și nu sînt șterși de o comandă inițializare-erori. De aceea, citirea stării indicatorilor de eroare, registrul RR1, va reflecta întotdeauna atît starea caracterului din *buffer*-ul de recepție cît și orice eroare de paritate sau depășire apărută de la ultima comandă inițializare-erori. Totodată, pentru a păstra corespondența între caracterele recepționate și indicatorii de eroare asociați, registrul de stare trebuie citit *înaintea* datelor, exceptînd situațiile ce vor fi descrise mai jos. Prin utilizarea întreruperilor vectorizate cerința se îndeplinește ușor, deoarece SIO generează un vector de întrerupere special, în cazul apariției unei erori sau a sfîrșitului de cadru.

Este posibil ca în starea citită după date să fie incluse și erorile următorului caracter, în cazul cînd acesta a fost recepționat. În situația în care operațiile de citire se execută suficient de rapid, suprapunerea indicatorilor de stare nu mai apare. O excepție intervine dacă se lucrează în modul întrerupere-la-primul-caracter-recepționat deoarece, în acest context, o întrerupere generată de o condiție specială face ca erorile și caracterul, chiar dacă au fost deja citite, să fie păstrate pînă la o comandă inițializare-erori. Astfel se previne recepția unor caractere noi pînă la trimiterea unei comenzi de inițializare. De asemenea, selectînd modul întrerupere-la-primul-caracter vectorul de întrerupere emis de SIO într-un ciclu de achitare va fi modificat de apariția unei erori. La depășire, ultimul caracter se încarcă peste caracterul recepționat anterior, care în acest fel se pierde. În același timp se poziționează indicatorul eroare-depășire-recepție și se modifică vectorul pe starea condiție-de-recepție-specială, dacă a fost în prealabil validat modul de lucru starea-afectează-vectorul.

---

* vezi § 1.4.5.2.

** vezi § 1.4.5.1.

Cînd se lucrează în modul *polling*, pentru ca UC să preia un caracter recepționat, va fi necesară testarea bitului caracter-recepționat-disponibil, D_0, din registrul RR0. Acest bit este pus pe „0" de logica internă a lui SIO cînd *buffer*-ele de recepție sînt goale. La transmisie interesează bitul *buffer*-transmisie-gol, D_2 din RR0, pus pe „1" de SIO ori de cîte ori *buffer*-ul de transmisie nu are nici un caracter. Înaintea fiecărei scrieri în acest *buffer* UC trebuie să testeze bitul D_2 din RR0, pentru a se asigura că nu apare o suprapunere de caractere.

1.4.4.3. MODURI DE TRANSMISIE SINCRONE

SIO-Z80 permite implementarea atît a protocoalelor orientate pe caracter, BCP, cît și a celor orientate pe bit, BOP. Protocoalele BCP sînt de tip MONOSYNC, cu un singur caracter de sincronizare de 8 biți, BISYNC, cu secvență de sincronizare de 16 biți, sau pot fi cu sincronizare externă. Caracterele de sincronizare se detectează fără a fi necesară întreruperea UC. SIO poate detecta și caractere de sincronizare de 5, 6 sau 7 biți. La BCP verificarea CRC se întîrzie cu un caracter, astfel încît UC poate invalida calculul CRC pentru anumite caractere, ceea ce permite implementarea unor protocoale ca BISYNC al firmei IBM. Circuitul poate calcula resturi utilizînd două polinoame generatoare: CRC-16, $x^{16} + x^{15} + x^2 + 1$, și CCITT, $x^{16} + x^{12} + x^5 + 1$. În modul SDLC, inițializarea se face cu „1", iar în celelalte cu „0". Dacă nu mai sînt date de transmis, SIO poate transmite automat caracterele CRC, astfel creîndu-se posibilitatea folosirii circuitului în transmisii de viteză mare prin DMA, fără intervenția UC la sfîrșitul unui mesaj. Cînd nu sînt nici date nici caractere CRC disponibile, SIO inserează caractere SYNC de 8 sau 16 biți, indiferent de lungimea programată a caracterelor.

Protocoalele BOP, cum sînt SDLC sau HDLC, se implementează comod cu SIO-Z80, circuitul asigurînd transmiterea automată a delimitatorului, inserarea și eliminarea biților „0", generarea CRC. Printr-o comandă specială se poate emite secvența ABORT. La sfîrșitul mesajului, cînd *buffer*-ul de transmisie devine gol, circuitul transmite automat caracterele CRC și delimitatorul. Dacă apare o eroare de ritm în timpul transmisiei mesajului, o întrerupere externă/de stare poate avertiza UC, aceasta avînd posibilitatea să comande generarea unei secvențe ABORT. Se pot transmite caractere de $1 \div 8$ biți și se pot recepționa mesaje, fără a se fi precizat în prealabil structura caracterelor în interiorul cîmpului de informație al unui cadru.

La recepția de tip BOP circuitul se sincronizează automat pe delimitatorul de început al unui cadru, generînd un semnal de sincronizare la conexiunea externă $\overline{SYNC}$ sau, dacă a fost programat, o întrerupere. SIO mai poate fi programat cu ajutorul cîmpului de adresă, pentru a căuta cadre adresate unui utilizator specificat, sau cadre emise către toți utilizatorii. Cadrele care nu îndeplinesc nici una din cele două condiții vor fi ignorate. Numărul de caractere din cîmpul de adresă poate fi extins prin program. Dacă se lucrează în întreruperi, se pot selecta cele două moduri amintite mai sus: întrerupere-la-primul-caracter-recepționat sau întrerupere-la-fiecare-caracter-recepționat.

Toate modurile sincrone impun folosirea unei rate de transfer a datelor egale cu frecvența ceasului, atît la transmisie cît și la recepție. Datele vor fi eșantionate la recepție cu frontul pozitiv al ceasului $\overline{RxC}$, în timp ce la trans-

misie schimbările de biți vor apărea pe frontul negativ al ceasului $\overline{TxC}$. După o inițializare hardware sau software circuitele de recepție se vor afla în modul așteptare-sincronizare, *hunt*, activat de fapt după o comandă validare-recepție. Transferul datelor va începe numai după realizarea sincronizării. La pierderea sincronizării se poate reintra în modul așteptare-sincronizare prin programarea pe „1" a bitului D_4 din registrul de comandă WR3. Diferențele între protocoalele de tip MONOSYNC, BISYNC și cu sincronizare externă sînt date numai de maniera în care se face sincronizarea; modul de funcționare trebuie să fie selectat înaintea încărcării caracterelor de sincronizare, deoarece registrele sînt folosite diferențiat:

— în modul MONOSYNC, caracter SYNC de 8 biți, validarea transferului de date se va face după detecția unui singur caracter SYNC programat în registrul WR7;

— în modul BISYNC, sincronizare pe 16 biți, asamblarea caracterelor va începe după detecția a două caractere de sincronizare adiacente programate în registrele WR6 și WR7;

— în modul cu sincronizare externă, asamblarea caracterelor va începe cu primul front pozitiv al ceasului $\overline{RxC}$, după activarea conexiunii externe $\overline{SYNC}$ care va trebui menținută pe „0" cel puțin trei perioade de ceas.

În modurile MONOSYNC și BISYNC conexiunea externă $\overline{SYNC}$ e folosită ca ieșire și se va activa ori de cîte ori va fi detectată o secvență de sincronizare, fiind ținută „0" pe durata perioadei de ceas în care s-a făcut detecția. În toate cele trei moduri asamblarea caracterelor va continua pînă la inițializarea circuitului SIO, pînă la invalidarea recepției, printr-o comandă sau cu ajutorul semnalului $\overline{DCD}$, dacă se lucrează cu auto-validare, sau pînă cînd UC programează modul așteptare-sincronizare.

1.4.4.3.1. Transmisia sincronă de tip BCP

După o inițializare sau cînd transmiterea nu este încă validată, circuitul SIO-Z80 menține linia de emisie în starea MARK. La programarea bitului transmisie-BREAK, D_4 din WR5, această linie va fi trecută în starea SPACE, indiferent de conținutul registrului de transmisie. În urma validării transmisiei și a selecției modului, în starea de repaus, circuitul va transmite continuu pe linie caracterul de sincronizare programat, de 8 sau 16 biți.

În transmisie, SIO poate fi utilizat în întreruperi sau în modul *polling*. Dacă se folosesc, întreruperile pot fi de transmisie sau externe/de stare. Programînd bitul validare-întreruperi-transmisie, D_1 din WR1, SIO va genera o întrerupere de fiecare dată cînd *buffer*-ul de transmisie se golește. Această întrerupere poate fi achitată fie prin scrierea unui nou caracter în *buffer*, fie cu ajutorul comenzii inițializare-întrerupere-transmisie-in-așteptare, comanda 5. Cînd întreruperea se achită cu această comandă și dacă nu se mai scrie nici un caracter în *buffer*, SIO nu va mai genera o altă întrerupere de transmisie. După scrierea unui caracter, *buffer*-ul se poate goli din nou, generîndu-se astfel o întrerupere de transmisie. Dacă se programează bitul validare-întreruperi-externe/de stare, D_0 din WR1, întreruperea se activează în situațiile: începerea transmisiei caracterelor CRC sau a caracterelor SYNC, schimbări de stare la intrările $\overline{DCD}$, $\overline{SYNC}$ și $\overline{CTS}$. Aceste tipuri de întrerupere pot avea

asociate un vector unic, în situația în care a fost în prealabil selectat modul de lucru starea-afectează-vectorul, D_2 din WR1.

Atunci cînd nu mai sînt date de transmis, SIO va insera automat caractere SYNC, dacă transmisia CRC nu a fost validată. Generarea unei întreruperi se va face numai după ce s-a încărcat primul dintre caracterele SYNC inserate. Dacă transmisia CRC este validată, în momentul în care nu mai sînt date de emis, SIO va transmite caracterele CRC, 16 biți urmate de caractere SYNC. Pe timpul cît se transmite restul CRC, indicatorul eroare-de-ritm-la-transmisie/EOM, D_6 în RR0, se pune pe „1", bitul buffer-transmisie-gol rămînînd pe „0", pentru a semnifica starea *buffer*-ului încă plin. CRC nu se calculează pentru caracterele SYNC inserate automat, dar va fi calculat pentru orice caracter SYNC emis ca date, exceptînd situația în care generatorul CRC este invalidat la încărcarea caracterului din *buffer* în registrul de deplasare pentru transmisie. După transmiterea caracterelor CRC, indicatorul buffer-transmisie-gol trece pe „1", generîndu-se și o întrerupere ce semnalează unității centrale posibilitatea de a începe un nou mesaj. Comanda generatorului CRC se poate face ținînd cont că, înainte de încărcarea datelor, va trebui inițializat prin trimiterea unei comenzi inițializare-generator-CRC-la-transmisie; datele pot fi încărcate numai după validarea generării CRC și a transmisiei. Înainte de a se emite caracterele CRC, dar, cel mai devreme după încărcarea primului caracter, trebuie trimisă comanda inițializare-eroare-de-ritm-la-transmisie/EOM. Cît timp eroare-de-ritm-la-transmisie/EOM este pe „1" transmisia CRC e inhibată, circuitul SIO putînd să insereze în cadrul mesajelor, cînd nu mai sînt date disponibile, caracterul SYNC. La sfîrșitul mesajului însă, acest bit trebuie pus pe „0", pentru a se permite emisia CRC.

Invalidarea transmisiei pe timpul emisiei unui caracter conduce, după transmiterea normală a caracterului, la trecerea liniei de emisie în starea MARK. De asemenea, dacă există un caracter în *buffer*, caracterul se păstrează. Pe de altă parte, la trimiterea de către UC a unei comenzi transmisie-BREAK, secvența BREAK, trecerea liniei în starea SPACE, va fi imediat emisă, caracterele în curs de transmisie pierzîndu-se.

1.4.4.3.2. Recepția sincronă de tip BCP

Validarea recepției se face după programarea modului și transmiterea caracterelor SYNC, în această ordine. În continuare, circuitele de recepție se vor găsi în starea așteptare-sincronizare în care vor rămîne pînă la: detecția unui caracter SYNC, în modul MONOSYNC, detecția a două caractere SYNC, în modul BISYNC, sau activarea pe „0" a conexiunii externe $\overline{\text{SYNC}}$. În primele două cazuri această conexiune funcționează ca ieșire pentru a indica sincronizarea, iar în ultimul caz ca intrare.

Asamblarea caracterelor începe imediat, după direcția secvenței de sincronizare. La recepție, ca și la emisie, SIO poate lucra cu sau fără întreruperi. Dacă se lucrează în întreruperi, acestea pot fi întrerupere-la-primul-caracter-recepționat sau întrerupere-la-fiecare-caracter-recepționat. Primul tip de întrerupere se utilizează, de obicei, pentru a lansa fie o buclă de testare software, fie o instrucțiune I/E de transfer pe bloc, întrebuințînd în ultimul caz ieșirea $\overline{\text{W/RDY}}$ pentru sincronizarea UC cu SIO. De asemenea, o astfel de întrerupere poate fi folosită și pentru a lansa un circuit specializat de acces

direct la memorie, DMA. Lucrînd cu acest tip de întrerupere, SIO-Z80 va întrerupe la primul caracter recepţionat şi, apoi, numai în caz de eroare. Anularea acestui mod de întrerupere se face cu ajutorul comenzii 4, iniţializare-întrerupere-la-primul-caracter-recepţionat. Al doilea tip de întrerupere face ca SIO să activeze linia $\overline{INT}$ ori de cîte ori în *buffer*-ul de recepţie se găseşte un caracter asamblat. Condiţiile speciale de recepţie sau erorile pot genera un vector unic, dacă în prealabil a fost selectat modul starea-afectează-vectorul. Eroarea de paritate poate, opţional, să nu conducă la generarea unui vector special.

Calculul CRC pentru un anumit caracter începe cu o întîrziere de 8 cicli de ceas după ce caracterul a fost transferat din registrul de deplasare în stiva de recepţie (a se vedea şi figura 1.49). Validarea/invalidarea CRC înaintea transferării unui caracter în stivă face ca precedentul caracter transferat în stivă să fie inclus/exclus din calculul CRC. Astfel, verificarea CRC poate fi comandată selectiv, existînd posibilitatea ca anumite caractere să nu intre în calculul CRC.

1.4.4.3.3. Transmisia sincronă de tip BOP (SDLC)

Starea de repaus a liniei de emisie, înainte de a valida transmisia, este MARK. După validarea transmisiei, pe această linie, în starea de repaus, se vor genera în mod continuu delimitatoare. Pentru a transmite un bloc de date este necesară mai întîi iniţializarea generatorului CRC cu ajutorul comenzii iniţializare-generare-CRC-la-transmisie. Această iniţializare poate fi făcută oricînd după emisia caracterelor CRC corespunzătoare mesajului precedent. Ca şi în cazul BCP, pe durata transmisiei restului CRC indicatorul eroare-de-ritm-la-transmisie/EOM se pune pe „1" iar *buffer*-transmisie-gol rămîne pe „0". La sfîrşitul transmisiei CRC, al doilea indicator se poziţionează şi el pe „1", conducînd la generarea unei întreruperi. Secvenţa CRC se transmite automat atunci cînd transmiţătorul nu mai are date de emis. După transmisia ei pe linie se vor emite delimitatoare. Generarea unei secvenţe ABORT se face prin comanda 1, transmisie-ABORT. Aceasta va conduce la generarea a $8 \div 14$ biţi „1", după care se va trece în starea de repaus emiţîndu-se în mod continuu delimitatoare. Caracterele în curs de transmisie sau în aşteptare în *buffer* se vor pierde. Inserarea de biţi „0" se face automat de către SIO după întîlnirea a cinci biţi „1" succesivi, exceptînd secvenţele delimitator şi ABORT.

Se pot transmite caractere de $1 \div 8$ biţi, numărul de biţi/caracter putînd fi schimbat pe timpul transmisiei unui bloc. În felul acesta se asigură ca lungimea cîmpului de date, I, dintr-un cadru, să fie variabilă la nivel de bit. Utilizînd codul-de-rest-I-la-recepţie, SIO are posibilitatea să recepţioneze un mesaj avînd orice lungime în biţi şi să-l retransmită exact cum l-a primit fără să fie necesare, în prealabil, alte informaţii despre structura caracterelor din cîmpul I. Schimbarea numărului de biţi/caracter nu afectează caracterul deplasat în registrul de transmisie, fiecare caracter fiind serializat cu numărul de biţi programat în momentul transferării din *buffer* în registrul de transmisie.

1.4.4.3.4. Recepția sincronă de tip BOP (SDLC)

Transferul de date între SIO și UC poate începe după primirea primului caracter diferit de un delimitator și după recepția cel puțin a unui delimitator, cu condiția să nu se lucreze în modul căutare-adresă, D_2 din WR3. Dacă se lucrează în acest mod, pentru lansarea transferului, mai este necesar ca, după delimitator, să urmeze fie adresa programată, fie adresa globală. Cînd nu se folosește modul căutare-adresă sau în cazul în care cîmpul de adresă este format din mai mulți octeți, există posibilitatea ca un mesaj nedorit să nu fie citit complet de UC. După luarea unei astfel de decizii de către UC, prin programarea bitului D_4 din WR3, intrare-în-modul-așteptare-sincronizare, SIO va suspenda recepția pînă la următorul delimitator de început al unui cadru.

Desfășurarea transferului poate fi comandată prin *polling* sau prin întreruperi. În prima situație, disponibilitatea unui caracter în *buffer*-ul de recepție se detectează cu ajutorul indicatorului caracter-recepționat-disponibil, D_0 din RR0. În a doua situație, prin selectarea întreruperii-la-primul-caracter-recepționat, se asigură o cale de începere a unui transfer de bloc. Pentru ieșirea din bucla de transfer, dacă nu se cunoaște lungimea cadrului, se poate folosi o întrerupere pe o condiție-de-recepție-specială, sfîrșit-de-cadru, generată la recepția delimitatorului de terminare a unui cadru. Pentru situațiile în care numărul de biți din cîmpul de date nu este un multiplu întreg al lungimii caracterelor, lungime programată cu biții D_7, D_6 din WR3, utilizatorul va trebui să țină cont și de valoarea codului-de-rest-I-la-recepție. Rearmarea întreruperii la primul caracter al următorului cadru se face prin transmiterea comenzii 4, inițializare-întrerupere-la-primul-caracter.

Biții „0", inserați la transmisie pentru a elimina din cîmpul de date secvențele-delimitator, sînt rejectați automat la recepție. Detectarea unei secvențe ABORT, mai mult de 7 biți „1", conduce la punerea pe „1" a indicatorului BREAK/ABORT din RR0 și generarea, dacă a fost validată, a unei întreruperi externe/de stare. Reactivarea acestei întreruperi se va face cu comanda 2, inițializare-întrerupere-externă/de-stare. Lungimea caracterelor poate fi modificată de UC chiar pe timpul recepției unui mesaj.

Validarea verificării CRC se face automat, calculul fiind inițializat cu delimitatorul de început al unui cadru și continuat pînă la întîlnirea delimitatorului final. Rezultatul verificării CRC este dat de indicatorul eroare-de-CRC/cadrare din RR1, „0" pentru mesaj corect recepționat. Verificarea de paritate poate fi utilizată pentru datele din cîmpul I, dacă acestea sînt reprezentate prin caractere de 5÷7 biți și se utilizează o legătură semi-duplex, neexistînd circuite separate pentru controlul parității la emisie și recepție.

1.4.5. PROGRAMAREA CIRCUITULUI SIO-Z80

Programarea circuitului SIO-Z80 impune transmiterea unor comenzi pentru selectarea modului de comunicație și, apoi, pentru stabilirea parametrilor specifici de lucru cu UC și modemul. Personalizarea funcționării circuitului înseamnă, în principal, scrierea registrelor de comandă.

1.4.5.1. REGISTRELE DE COMANDĂ

Pentru a fi programat, oricare din cele șapte sau opt registre de comandă ale canalului A, respectiv B, necesită scrierea de către UC a doi octeți. Excepție face numai registrul WR0. O inițializare hardware sau software va poziționa logica de adresare a registrelor pe WR0 astfel încît primul octet de comandă transmis va fi înscris în acest registru. De aceea, WR0 conține principalele comenzi și cîmpul de inițializare CRC. De asemenea, biții D_2, D_1 și D_0 din WR0 specifică adresa registrului în care se va înscrie octetul de comandă următor.

Registrul de comandă WR0 are următorul format:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Cod inițializare CRC		Comandă			Adresă registru următor		

Adresa registrului următor este numărul registrului exprimat în binar cu ajutorul biților D_2, D_1, D_0.

Cele 8 comenzi care se pot transmite unui canal SIO sînt codificate conform tabelului:

Comanda	D_5	D_4	D_3	Denumirea comenzii
0	0	0	0	Comandă-neoperantă
1	0	0	1	Transmisie-ABORT (modul SDLC)
2	0	1	0	Inițializare-întreruperi-externe/de-stare
3	0	1	1	Inițializare-canal
4	1	0	0	Inițializare-întrerupere-la-primul-caracter-recepționat
5	1	0	1	Inițializare-întrerupere-transmisie-în-așteptare
6	1	1	0	Inițializare-erori
7	1	1	1	Revenire-din-întrerupere (numai canalul A)

Comanda 0, comandă-neoperantă, nu are nici un efect, fiind folosită la transmiterea adresei registrului următor.

Comanda 1, transmisie-ABORT, utilizată numai în procedura SDLC, pentru a iniția generarea unei secvențe de 8 ÷ 14 biți „1".

Comanda 2, inițializare-întreruperi-externe/de-stare, validează o nouă poziționare a biților de stare din RR0, după generarea unei întreruperi externe/de stare, care blocase acești biți pentru a-i menține stabili pînă la citirea lor de către UC.

Comanda 3, inițializare-canal, realizează aceeași inițializare ca în cazul activării intrării $\overline{\text{RESET}}$, dar numai pe un canal. Trimisă pe canalul A, inițializează și logica de priorități a întreruperilor. După transmiterea acestei comenzi, pînă la următoarea comandă, este necesară trecerea a cel puțin 4 cicli de ceas.

Comanda 4 inițializare-întrerupere-la-primul-caracter-recepționat [reactivează acest tip de întrerupere, dacă a fost selectat, pregătind recepția următorului mesaj.

Comanda 5, inițializare-întrerupere-transmisie-în-așteptare, previne generarea întreruperilor de transmisie, condiționate de golirea *buffer*-ului, după

terminarea transmiterii datelor, pînă la scrierea de către UC a unui nou caracter în *buffer*-ul de transmisie.

Comanda 6, inițializare-erori, șterge eventualele erori de paritate și/sau depășire memorate în RR1 de la precedenta inițializare.

Comanda 7, revenire-din-întrerupere, trimisă numai pentru canalul A, este interpretată de SIO ca o instrucțiune RETI conducînd la ștergerea bistabilului „Întrerupere în curs de tratare", ceea ce permite validarea întreruperilor de la dispozitivele cu prioritate scăzută. Existența acestei comenzi asigură folosirea priorităților interne din SIO și în sistemele care nu se bazează pe familia Z80.

Codul inițializare CRC, D_7, D_6, are semnificația din tabelul de mai jos:

D_7	D_6	Denumirea comenzii
0	0	Comandă-neoperantă
0	1	Inițializare-verificare-CRC-la-recepție
1	0	Inițializare-generare-CRC-la-transmisie
1	1	Inițializare-eroare-de-ritm-la-transmisie/EOM

Registrul de comandă WR1 are următorul format:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Validare WAIT/ READY	WAIT/ READY	W/RDY la recepție/transmisie	Mod de întrerupere la recepție		Starea afectează vectorul (numai canal B)	Validare întreruperi transmisie	Validare întreruperi externe/de stare

Validare-întreruperi-externe/de-stare activează generarea întreruperilor, ca urmare a unor tranziții la intrările $\overline{DCD}$, $\overline{CTS}$ și $\overline{SYNC}$, a întîlnirii unei secvențe BREAK sau la începutul transmiterii caracterelor CRC ori SYNC. Dacă intrările enumerate nu se folosesc, ele vor trebui conectate la +5V, pentru a se preveni apariția unor întreruperi false.

Validare-întreruperi-transmisie permite activarea întreruperii ori de cîte ori *buffer*-ul de transmisie devine gol.

Starea-afectează-vectorul, bit programabil numai pentru canalul B, care, dacă e pus pe „1", permite selectarea facilității ca vectorul trimis de SIO într-un ciclu de achitare-întrerupere să fie modificat conform tabelului:

V_3	V_2	V_1	Semnificația modificării	
0	0	0	Buffer-transmisie-gol	Canal B
0	0	1	Schimbare-externă/de-stare	
0	1	0	Caracter-recepționat-disponibil	
0	1	1	Condiție-de-recepție-specială	
1	0	0	Buffer-transmisie-gol	Canal A
1	0	1	Schimbare-externă/de-stare	
1	1	0	Caracter-recepționat-disponibil	
1	1	1	Condiție-de-recepție-specială	

Dacă $D_2=0$ vectorul trimis de SIO este cel scris prin program în registrul WR2.

Selectarea *modului-de-întrerupere-la-recepţie* se face cu ajutorul biţilor D_4 şi D_3:

D_4	D_3	Mod de întrerupere la recepţie
0	0	Întrerupere-recepţie-invalidată
0	1	Întrerupere-la-primul-caracter-recepţionat
1	0	Întrerupere-la-fiecare-caracter-recepţionat/Paritatea-afectează-vectorul
1	1	Întrerupere-la-fiecare-caracter-recepţionat/Paritatea-nu-afectează-vectorul

$\overline{W}$/RDY-la-recepţie/transmisie selectează, dacă ieşirea $\overline{W/RDY}$ este validată, situaţia în care această conexiune externă va fi activată: cînd *buffer*-ul de recepţie este gol, $D_5 = 1$, sau cînd *buffer*-ul de transmisie este plin, $D_5=0$.

WAIT/READY se programează pe „0", dacă linia $\overline{W/RDY}$ se utilizează pentru a comanda o intrare de tip WAIT a unui microprocesor, şi pe „1" dacă se conectează la o intrare de tip READY a unui circuit specializat pentru DMA. Funcţia READY poate fi realizată de SIO oricînd, nefiind condiţionată de selectarea circuitului, în timp ce funcţia WAIT este activă numai cînd UC încearcă să citească date din SIO, în momentul cînd *buffer*-ul de recepţie este gol, sau să scrie cînd *buffer*-ul de transmisie e plin.

Cît timp bitul *validare-WAIT/READY* e menţinut pe „0" ieşirea $\overline{W/RDY}$ va fi „1" pentru modul READY şi în starea a treia pentru modul WAIT. $D_7=1$ va însemna validarea funcţionării ieşirii $\overline{W/RDY}$ într-unul din cele două moduri.

Registrul de comandă WR2 este registrul care memorează vectorul de întrerupere al circuitului şi nu există decît în setul de registre al canalului B. Într-un ciclu de achitare a întreruperii, SIO va returna un vector avînd biţii $V_7 \div V_4$ şi V_0 la fel cu cei programaţi şi dacă a fost programat bitul starea-afectează-vectorul, cu V_3, V_2, V_1 modificaţi ca în tabelul de mai sus*. Formatul acestui registru este:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
V_7	V_6	V_5	V_4	V_3	V_2	V_1	V_0
Se returnează nemodificaţi				Modificaţi numai dacă D_2/WR1= 1 (canal B)			Nemodificat

Registrul de comandă WR3 conţine comenzi şi parametri referitori la logica de recepţie:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Lungime caracter la recepţie		Auto-validare	Aşteptare sincronizare	Validare CRC la recepţie	Căutare adresă	Invalidare încărcare caracter SYNC	Validare recepţie

* vezi structura registrului WR1 . // .

Validare-recepție pe „1" permite începerea operațiilor de recepție.

Programînd *invalidare-încărcare-caracter-SYNC*, caracterele SYNC ce preced un mesaj nu se vor mai transfera în *buffer*-ul de recepție.

Prin selecția modului *căutare-adresă*, dacă se lucrează în protocolul SDLC, SIO va rejecta toate cadrele care nu au adresa egală cu cea programată sau cu adresa globală de emisie.

Punerea pe „1" a bitului *validare-CRC-la-recepție* conduce la o reluare sau inițiere a calculului CRC începînd cu ultimul caracter transferat din registrul de recepție în stivă, fără a se mai ține cont de eventualele caractere precedente păstrate în stivă.

Programarea pe „1" a bitului *așteptare-sincronizare* permite reintrarea în modul de așteptare a secvenței de sincronizare în cazurile cînd s-a pierdut sincronizarea sau, într-un protocol de tip BOP, cînd UC consideră că mesajul primit nu este necesar.

Selectînd modul *auto-validare*, $D_5=1$, intrările $\overline{DCD}$ și $\overline{CTS}$ vor fi folosite pentru validarea circuitelor de recepție, respectiv transmisie. $D_5=0$ inhibă această funcție menținînd doar posibilitatea ca starea intrărilor să fie citită în registrul RR0.

Lungimea caracterelor asamblate la recepție este determinată de valoarea biților D_7, D_6 conform tabelului:

D_7	D_6	Număr de biți/caracter
0	0	5
0	1	7
1	0	6
1	1	8

Numărul de biți/caracter poate fi reprogramat în timpul asamblării.

Registrul de comandă WR4, conținînd comenzi ce afectează atît transmisia cît și recepția, are următorul format:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Rata datelor		Moduri de sincronizare		Număr de biți STOP		Paritate pară/impară	Paritate

Dacă *paritate* este pus pe „1" SIO adaugă la numărul de biți specificați în D_7, D_6/WR3, bitul de paritate. De asemenea, dacă $D_0=1$, la recepție se ia în considerare și bitul de paritate.

Paritate-pară/impară specifică atît pentru transmisie cît și pentru recepție felul parității: $D_1=0$, paritate impară, $D_1=1$, paritate pară.

Numărul-de-biți-STOP adăugați la transmisia asincronă a caracterelor se fixează cu ajutorul biților D_3 și D_2:

D_3	D_2	Număr de biți STOP
0	0	Moduri sincrone
0	1	1 bit STOP pe caracter
1	0	$1^1/_2$ biți STOP pe caracter
1	1	2 biți STOP pe caracter

Selecția *modului-de-sincronizare* se face cu D_5, D_4:

D_5	D_4	Moduri de sincronizare
0	0	Caracter SYNC de 8 biți transmis prin program
0	1	Caracter SYNC de 16 biți transmis prin program
1	0	Mod SDLC (secvența de sincronizare 01111110)
1	1	Sincronizare externă

Biții D_7, D_6 se utilizează pentru specificarea ratei de transmisie a datelor, în funcție de frecvența ceasului de emisie/recepție. Pentru modurile sincrone rata datelor trebuie să fie aleasă egală cu frecvența ceasului, în timp ce pentru modurile asincrone se poate selecta orice raport. Raportul ales va fi folosit de SIO atît pentru emisie cît și pentru recepție. Precizăm că pentru oricare mod de lucru se va avea în vedere ca ceasul sistemului, Φ, să aibă frecvența de cel puțin 5 ori mai mare decît rata datelor. De asemenea, pentru o rată a datelor egală cu frecvența ceasului sincronizarea de bit trebuie asigurată extern. Selecția *ratei datelor* se face conform tabelului:

D_7	D_6	Rata datelor
0	0	Rata datelor × 1 = Frecvența ceasului
0	1	Rata datelor × 16 = Frecvența ceasului
1	0	Rata datelor × 32 = Frecvența ceasului
1	1	Rata datelor × 64 = Frecvența ceasului

Registrul de comandă WR5 conține cea mai mare parte a comenzilor și parametrilor ce afectează transmisia:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
DTR	Lungime caracter la transmisie		Transmisie BREAK	Validare transmisie	CCITT/ CRC-16	RTS	Validare CRC la transmisie

Validare-CRC-la-transmisie permite, dacă este pus pe „1", transmisia automată a caracterelor CRC atunci cînd nu mai sînt date de emis. Cu ajutorul acestui bit de comandă se poate valida/invalida calculul CRC pentru orice caracter.

RTS comandă ieșirea $\overline{RTS}$: $D_1=1$ conduce la activarea, trecerea pe „0", a ieșirii $\overline{RTS}$ în timp ce $D_1=0$ permite trecerea pe „1" a ieșirii $\overline{RTS}$ dar numai după ce transmițătorul este gol.

CCITT/CRC-16 selectează polinomul generator utilizat în verificarea CRC, atît la emisie, cît și la recepție: „0" pentru polinomul CCITT, $x^{16} + x^{12} + x^5 + 1$.

Cît timp bitul *validare-transmisie* este „0" ieșirea TxD va fi ținută în starea MARK, „1". Emisia va începe după trecerea bitului pe „1". Punerea pe „0" a acestei comenzi va încheia transmisia după terminarea caracterului curent sau imediat, dacă invalidarea se face pe timpul emisiei unui caracter CRC.

Poziţionarea pe „1" a comenzii *transmisie-BREAK* forţează linia TxD în starea SPACE, „0".

Numărul de biţi dintr-un octet transferat în *buffer*-ul de transmisie, ce vor fi deci emişi în linie, este selectat cu ajutorul biţilor D_6 şi D_5:

D_6	D_5	Număr de biţi/caracter
0	0	5 sau mai puţini
0	1	7
1	0	6
1	1	8

Biţii care se emit se presupun aliniaţi la dreapta octetului, transmisia începînd cu bitul cel mai puţin semnificativ, D_0. În cazul transmiterii a 5 sau mai puţini biţi structura octetului va fi:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Număr biţi/caracter
1	1	1	1	0	0	0	D	1 bit
1	1	1	0	0	0	D	D	2 biţi
1	1	0	0	0	D	D	D	3 biţi
1	0	0	0	D	D	D	D	4 biţi
0	0	0	D	D	D	D	D	5 biţi

DTR este bitul de comandă al ieşirii $\overline{DTR}$: $D_7=1$ conduce la activarea, punerea pe „0", a ieşirii DTR, $D_7=0$ face ca $\overline{DTR}=1$.

Registrul de comandă WR6 poate conţine primii 8 biţi ai unui caracter de sincronizare de 16 biţi (BISYNC), o adresă, dacă se utilizează modul căutare-adresă, sau caracterul SYNC dacă a fost selectat modul de sincronizare pe caracter SYNC de 8 biţi:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
$SYNC_7$	$SYNC_6$	$SYNC_5$	$SYNC_4$	$SYNC_3$	$SYNC_2$	$SYNC_1$	$SYNC_0$
AD_7	AD_6	AD_5	AD_4	AD_3	AD_2	AD_1	AD_0

Registrul de comandă WR7 poate conţine al doilea octet al caracterului de sincronizare de 16 biţi, caracterul de sincronizare de 8 biţi sau, pentru protocolul SDLC, secvenţa 01111110. Cele două registre, WR6 şi WR7, nu se utilizează în modul de sincronizare externă. Formatul lui WR7 este:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
$SYNC_{15}$	$SYNC_{14}$	$SYNC_{13}$	$SYNC_{12}$	$SYNC_{11}$	$SYNC_{10}$	$SYNC_9$	$SYNC_8$
0	1	1	1	1	1	1	0

1.4.5.2. REGISTRELE DE STARE

SIO-Z80 are, pentru memorarea stării fiecărui canal, cîte trei registre de stare: RR0, RR1 și RR2. RR2, vectorul de întrerupere modificat, nu poate fi citit decît pe canalul B. Citirea unui registru de stare presupune două operații de I/E: întîi scrierea adresei registrului în WR0 și apoi citirea propriu-zisă.

Registrul de stare RR0 are următorul format:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
BREAK/ ABORT	Eroare de ritm la transmisie/ EOM	CTS	SYNC/așteptare sincronizare	DCD	*Buffer* transmisie gol	Întrerupere în așteptare (numai canal A)	Caracter recepționat disponibil

Caracter-recepționat-disponibil este pus pe „1" cînd în *buffer*-ul de recepție se află cel puțin un caracter.

Indicatorul *întrerupere-în-așteptare*, accesibil numai pe canalul A, fiind întotdeauna „0" pe canalul B, reflectă starea întreruperilor pentru întregul circuit.

Buffer-transmisie-gol se pune pe „1" ori de cîte ori *buffer*-ul de transmisie devine gol, cu excepția situației în care se transmit caracterele CRC în modurile sincrone.

DCD indică starea intrării $\overline{DCD}$ în timpul ultimei schimbări a oricărui indicator extern/de stare (DCD, CTS, SYNC/așteptare-sincronizare, BREAK/ABORT, eroare-de-ritm-la-transmisie/EOM). Pentru a obține starea curentă a intrării $\overline{DCD}$, bitul DCD va trebui citit imediat după trimiterea comenzii 2, inițializare-întreruperi-externe/de-stare.

SYNC/așteptare-sincronizare indică, în modurile asincrone, starea intrării SYNC. În modurile sincrone, bitul se pune pe „0" în momentul sincronizării și se poziționează pe „1" prin programarea bitului așteptare-sincronizare din WR3.

CTS indică starea conexiunii externe $\overline{CTS}$.

Eroare-de-ritm-la-transmisie/EOM pe „1" permite generarea întreruperii dacă a fost validată, la o transmisie într-un mod sincron, în momentul emisiei automate a caracterelor CRC, cînd *buffer*-ul de transmisie devine gol. Această întrerupere nu va fi generată dacă bitul D_6 e pe „0". $D_6=1$ și $D_2=0$ indică emisia unui caracter CRC, iar $D_6=1$ și $D_2=1$ indică emisia caracterelor SYNC.

BREAK/ABORT se pune pe „1" la detecția unei secvențe BREAK în modurile asincrone. După ce biții de stare vor fi inițializați prin trimiterea comenzii 2, bitul D_7 va putea trece pe „0" la terminarea secvenței BREAK. Dacă întreruperile externe/de stare sînt validate, schimbarea stării bitului D_7 va conduce la generarea întreruperii. În modul SDLC, D_7 se va pune pe „1" la detecția unei secvențe ABORT (cel puțin 7 biți „1").

Registrul de stare RR1 are formatul dat în continuare. După citirea acestui registru zona de adresă a registrului următor din WR0 se va pune pe 000.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Sfîrșit de cadru (SDLC)	Eroare de CRC/cadrare	Eroare depășire recepție	Eroare paritate	Codul de rest I la recepție			Toate caracterele transmise

Toate-caracterele-transmise e pus pe „1" în modurile asincrone, la transmisia completă a ultimului caracter din *buffer*-ul de transmisie. Modificarea acestui bit nu generează întrerupere. În modurile sincrone este întotdeauna „1".

Codul-de-rest-I-la-recepție indică lungimea cîmpului de date, I, dintr-un cadru, în situațiile în care aceasta nu e un multiplu întreg de lungimea programată a caracterului. Codul are semnificație nmai în cazul transferurilor în care s-a poziționat indicatorul sfîrșit-de-cadru (SDLC). Codul specifică biții de informație din ultimii doi octeți preluați de unitatea centrală, biții rămași din cîmpul I fiind întotdeauna aliniați la dreapta în cadrul unuia din cei doi octeți. Dăm mai jos valorile codului-de-rest-I-la-recepție în cazul în care numărul de biți/caracter a fost programat 8:

D_3	D_2	D_1	Biți I în ultimul octet	Biți I în penultimul octet
1	0	0	0	3
0	1	0	0	4
1	1	0	0	5
0	0	1	0	6
1	0	1	0	7
0	1	1	0	8
1	1	1	1	8
0	0	0	2	8

Atunci cînd limita ultimului caracter asamblat coincide cu limita cîmpului I, începutul caracterelor CRC, valoarea codului va fi 011. Pentru număr de biți/caracter diferiți de 8 se pot construi tabele asemănătoare cu cel de sus.

Eroare-paritate va fi pus pe „1" pentru acele caractere a căror paritate calculată nu este egală cu paritatea recepționată, evident cu condiția de a valida paritatea, D_0 în WR4. Această eroare se memorează pînă la trimiterea unei comenzi 6, inițializare-erori.

Eroare-depășire-recepție indică faptul că SIO a recepționat mai mult de patru caractere fără ca UC să fi citit vreunul. Această eroare se atașează numai caracterului înscris peste celelalte, dar, după citirea lui, ea se va memora pînă la trimiterea comenzii 6. Dacă s-a programat bitul starea-afectează-vectorul, încărcarea în stivă a caracterului care provoacă depășire va conduce la generarea unei întreruperi, cu vectorul corespunzînd condiției-de-recepție-speciale.

Eroare-de-CRC/cadrare se pune pe „1" în modurile asincrone, dacă apare o eroare de cadrare, numai pentru caracterul greșit, fără a se memora în continuare. La detecția unei erori de cadrare se adaugă, la durata unui caracter, un timp corespunzînd unei jumătăți de bit, pentru ca eroarea să nu fie interpretată ca un nou bit de START. În modurile sincrone, bitul D_6 indică rezultatul verificării CRC.

Sfîrșit-de-cadru (SDLC) indică în modul SDLC recepția unui delimitator de sfîrșit de cadru, eroare-de-CRC/cadrare și codul-de-rest-I-la-recepție devenind valide la poziționarea pe „1" a acestui bit.

Registrul de stare RR2, citit numai pe canalul B, conține vectorul de întrerupere, la fel cu cel care a fost scris în WR2, dacă bitul starea-afectează-vectorul e „0" sau modificat, cînd acest bit e „1". Dacă SIO nu are întreruperi în așteptare, vectorul are biții V_3, V_2 și V_1 egali cu 011, ceilalți fiind la fel cu cei programați în WR2. Formatul registrului RR2 este:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
V_7	V_6	V_5	V_4	V_3	V_2	V_1	V_0
La fel ca biții din WR2				Modificați dacă bitul starea-afectează-vectorul e „1"			La fel ca în WR2

1.4.5.3. UN EXEMPLU DE PROGRAMARE

În exemplul ce urmează se va presupune că SIO este selectat cu ajutorul bitului de adresă A_5 iar liniile C/$\overline{D}$ și B/$\overline{A}$ sînt comandate cu biții cei mai puțin semnificativi ai magistralei de adresă; adresele de I/E utilizate vor fi cele din tabelul:

Conexiuni externe			Adrese de I/E	Semnificația octetului transferat
$\overline{CE}(A_5)$	C/$\overline{D}$(A_1)	B/$\overline{A}$(A_0)		
0	0	0	20H	Date canal A
0	0	1	21H	Date canal B
0	1	0	22H	Comenzi/stări canal A
0	1	1	23H	Comenzi/stări canal B

Secvențele de program date în continuare programează canalul B să lucreze asincron, iar canalul A în protocolul SDLC. Secvențele urmează după o inițializare hardware sau software.

```
ASYNC:  LD  A,18H      ; Inițializare — canal B
        OUT (23H),A
        OUT (23H),A
```

```
        LD   A,02H     ; Adresare registru WR2/B
        OUT  (23H),A
        LD   A,80H     ; Încărcare vector întrerupere
        OUT  (23H),A
        LDA  A,04H     ; Adresare registru WR4/B
        OUT  (23H),A
        LD   A,47H     ; Rata datelor ×16 = Frecvenţa ceasului,
        OUT  (23H),A   ; 1 bit STOP, paritate pară
        LD   A,05H     ; Adresare registru WR5/B
        OUT  (23H),A
        LD   A,2AH     ; 7 biţi/caracter la transmisie,
        OUT  (23H,)A   ; validare-transmisie, activare ieşire RSTB
        LD   A,03H     ; Adresare registru WR3/B
        OUT  (23H),A
        LD   A,61H     ; 7 biţi/caracter la recepţie, auto-validare,
        OUT  (23H),A   ; validare-recepţie
        LD   A,01H     ; Adresare registru WR1/B
        OUT  (23H),A
        LD   A,17H     ; Întrerupere-la-fiecare-caracter-recepţionat/
        OUT  (23H),A   ; paritatea-afectează-vectorul, validare-între-
                       ; ruperi-transmisie, validare-întreruperi-
                       ; externe/de-stare

SDLC:   ...
        LD   A,18H     ; Iniţializare — canal A
        OUT  (22H),A
        OUT  (22H),A
        LD   A,04H     ; Adresare registru WR4/A
        OUT  (22H),A
        LD   A,20H     ; SDLC, Rata datelor×1=Frecvenţa ceasu-
        OUT  (22H),A   ; lui, fără paritate
        LD   A,46H     ; Adresare registru WR6/A, iniţializare-
        OUT  (22H),A   ; verificare-CRC-la-recepţie
        LD   A,ADR     ; Încărcare adresă mesaj SDLC
        OUT  (22H),A
        LD   A,87H     ; Adresare registru WR7/A, iniţializare-
        OUT  (22H),A   ; generare-CRC-la transmisie
        LD   A,7EH     ; Încărcare delimitator SDLC
        OUT  (22H),A
        LD   A,01H     ; Adresare registru WR1/A
        OUT  (22H),A
        LD   A,17H     ; Întrerupere-la-fiecare-caracter-recepţionat,
```

```
      OUT  (22H),A   ; starea-afectează-vectorul, validare-întreruperi-
                     ; transmisie,   validare-întreruperi-externe/de-
                     ; stare
      LD   A,15H     ; Adresare  registru  WR5/A,   inițializare-
      OUT  (22H),A   ; întreruperi-externe/de-stare
      LD   A,0E9H    ; DTR, 8 biți/caracter la transmisie, validare-
      OUT  (22H),A   ; transmisie, validare-CRC-la-transmisie
      LD   A,03H     ; Adresare WR3/A
      OUT  (22H),A
      LD   A,0EDH    ; 8  biți/caracter  recepționat,  auto-validare,
      OUT  (22H),A   ; validare-CRC-la-recepție, căutare-adresă, va-
      ...            ; lidare-recepție
```

BIBLIOGRAFIE

1. * * * Zilog, 1982/1983 *Data Book.*
2. * * * Mostek, 1979 *Microcomputer Data Book.*
3. * * * Mostek, *Z80 Microcomputer Software. Programming Guide.*
4. NICHOLS, E. A.; NICHOLS, J. C.; RONY, P. E., *Z-80, Livre 1, Programmation*, Publitronic Sarl, 1981.
5. NICHOLS, J. C.; NICHOLS, E. A.; RONY, P.E., *Z-80 Microprocessor, Programming & Interfacing, Book* 2, Howard W. Sams & Co., Inc., Indianopolis, 1979.
6. CARR, J. J., *Z80 Users Manual*, Reston Publishing Company, Inc., Reston, Virginia, 1980.
7. MILLER, A. R., *8080/Z80 Assembly Language: Techniques for Improved Programming*, John Wiley & Sons, 1981.
8. WAGNER, W., *Sistemul de microprocesoare U880D*, RFT.
9. MUREȘAN, T.; STRUNGARU, C.; STOINESCU, R.; PETRIU, E., *Microprocesorul 8080 în aplicații*, Editura Facla, Timișoara, 1981.
10. CĂPĂȚÎNĂ, O. D.; HAȘEGAN, M. C.; PUȘCĂ, M. V.. *Proiectarea cu microprocesoare*, Editura Dacia, Cluj-Napoca, 1983.
11. PETRESCU, A. (coordonator), *Microcalculatoarele Felix M18, M18B, M118*, Editura Tehnică, București, 1984.
12. PETRESCU, A. (coordonator), *Totul despre ... calculatorul personal aMIC*, Editura Tehnică, București, 1985.
13. TOACȘE, G., *Introducere în microprocesoare*, Editura Științifică și Enciclopedică, București, 1985.
14. WEISSBERGER, A. J., *Data Communications Handbook*, Signetics Corporation, 1979.
15. COCULESCU, L.; POINARIU, C., *Teleprelucrarea datelor*, Editura Militară, București, 1982.
16. DAVIES, D. W.; BARBER, D. L. A., *Rețele de interconectarea calculatoarelor*, Trad. din lb. engleză, după ediția a II-a, 1975, Editura Tehnică, București, 1976.
17. DAVIES, D. W.; BARBER, D. L. A.; PRICE, W. L.; SALMONIDES, C. M., *Computer Networks and Their Protocols*, John Wiley & Sons, 1979.
18. ROLLAND, B., *La liaison entre contrôleurs et imprimantes: les différentes interfaces*, Minis et micros, 1985, No. 240, p. 87–91.
19. * * * Intel, *Peripheral Design Handbook.*

20. * * * *Intel 8080 Microcomputer Systems User's Manual*, september 1975.
21. * * * Philips, *Data Handbook, Microprocessors, microcontrollers and peripherals*, Book IC14N, 1985.
22. NICHOLS, E. A.; NICHOLS, J. C.; MUSSON, K. R., *Data Communications for Microcomputers. Practical Experiments for Z80 Microcomputers*, McGraw-Hill, 1983.
23. DOLL, D. R., *Data Communications: Facilities, Networks and Systems Design*, John Wiley & Sons, 1978.
24. FOLTS, H. C., *McGraw-Hill Compilation of Data Communication Standards*, McGraw-Hill, 1981.
25. SEIDLER, J., *Principles of Computer Communication Network Design*, John Wiley & Sons, 1983.
26. LUPU, C.; ȚEPELEA, V.; PURICE, E., *Microprocesoare. Aplicații*, Editura Militară, București, 1982.

CONECTAREA MEMORIILOR EXTERNE CU DISCURI FLEXIBILE LA SISTEME CU MICROPROCESOARE

2.1. UNITĂŢI DE DISCURI FLEXIBILE — UDF

Introdusă ca mediu de înmagazinare şi transfer de date la un echipament de culegere de date, memoria externă cu discuri flexibile a cunoscut o surprinzătoare forţă de expansiune în lumea tehnicii de calcul, datorită acomodării sale cu noile inovaţii din electronica anilor de după 1970. Timpul de acces relativ mare, comparativ cu cel de la unităţile de disc obişnuite ale calculatoarelor puternice, s-a potrivit cu viteza de calcul relativ mică a microprocesoarelor, iar fiabilitatea destul de bună şi preţul de cost foarte redus au permis ca suportul utilizat la început pentru stocare să fie folosit şi in operaţiile curente, ca disc sistem sau disc de lucru. Sosită ca un înlocuitor al benzii de hîrtie perforată, discheta a mijlocit, după apariţia primelor sisteme de operare pe disc flexibil (1975), o rapidă dezvoltare a calculatoarelor cu preţ de cost redus, care au pătruns ulterior în cele mai variate domenii ale activităţii umane.

2.1.1. DEFINIŢII

Suportul magnetic încorporat în dischetă, *diskette*, are nevoie de un dispozitiv electromecanic pentru crearea condiţiilor ca interfaţa mediu—cap magnetic să faciliteze transferul de date între mediul magnetic şi o interfaţă electrică stabilită, de regulă cu niveluri logice „0" (0V÷0,8 V) şi „1" (2,4V÷÷5V). Acest dispozitiv, *driver*, prezentat in figura 2.1, este denumit în continuare *unitate de disc flexibil*, UDF, fiind un echipament periferic de sine stătător. Împreună cu blocul de cuplare la magistrala microcalculatorului, una sau mai multe UDF formează un *subsistem de disc flexibil*, SSDF, ca în figura 2.2. Adaptarea interfeţei UDF la magistrala microcalculatorului este realizată într-un modul care organizează datele de pe suport conform unui format standardizat sau specializat al pistei. Această parte se numeşte *formator*, FRM. Formatorul, împreună cu *circuitele de dialog cu magistrala*, CDM, se constituie într-o unitate numită *cuplor de disc flexibil*, CDF. În concluzie, un SSDF se compune din mai multe UDF şi un CDF, acesta din urmă fiind format din FRM şi CDM. Modul de organizare descris se întîlneşte şi la alte subsisteme ataşate unui calculator (bandă magnetică, disc de capacitate mare, staţie grafică etc.).

Dăm mai jos definiţiile altor termeni utilizaţi în capitolul de disc flexibil.

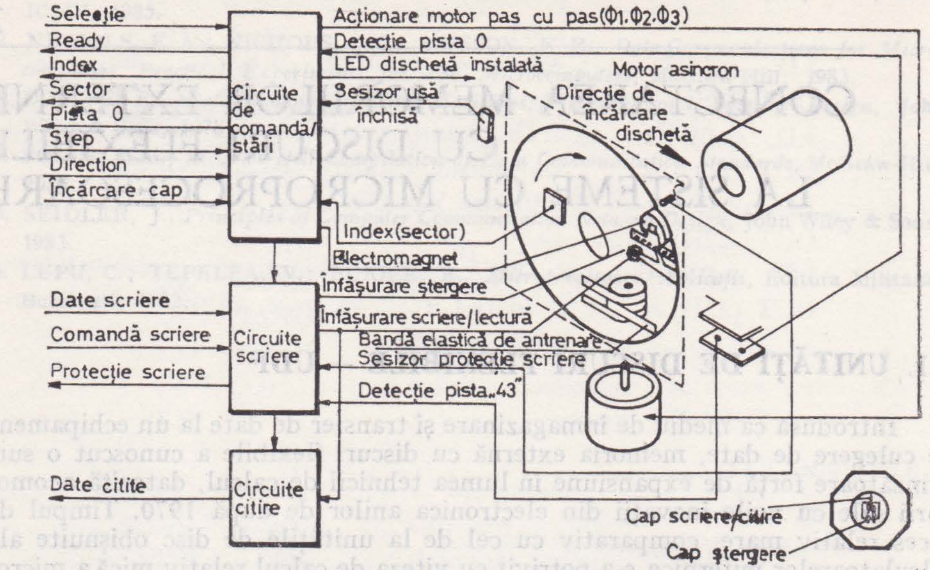


Fig. 2.1. Structura unei unități de disc flexibil

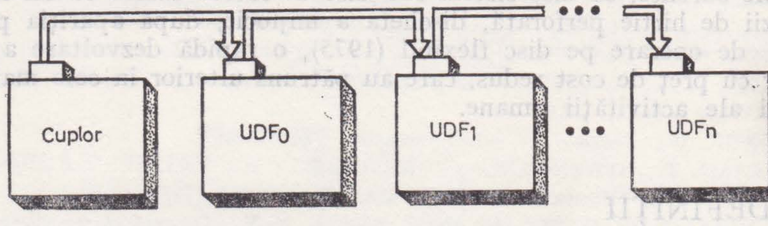


Fig. 2.2. Subsistem de disc flexibil

Mod de înregistrare: procedeu de reprezentare a informației, în vederea adaptării la caracteristicile mediului de înregistrare, prin care simbolurile de informație sînt înlocuite prin cuvinte de cod.

Modulație de frecvență, Frequency Modulation, *FM:* mod de înregistrare în care un simbol de informație se codează într-un cuvînt de cod cu două simboluri, *celula-bit,* astfel că primul simbol înregistrat, numit simbol de sincronizare — „ceas", este întotdeauna „1", ar al doilea simbol este chiar informația ce se dorește a fi memorată.

Modulație modificată a frecvenței, Modified Frequency Modulation, *MFM:* mod de înregistrare derivat din FM, în care simbolul de sincronizare dintr-un cuvînt de cod lipsește, dacă există simbol de informație adiacent, în cuvîntul precedent sau în celula curentă. Se obține o micșorare a interferenței intersimbol și o eficiență mai bună a codării.

Densitate de înregistrare (liniară) a informațiilor: raport între numărul de informații înregistrate și lungimea fizică a înregistrării. Se exprimă în *bits per inch,* bpi, sau biți pe mm, bit/mm, (1 bit/mm = 25,4 bpi).

Densitate de înregistrare a tranzițiilor: raport între numărul de tranziții în magnetizare înregistrate și lungimea fizică a înregistrării. Se măsoară în tranziții pe *inch,* frpi — *flux reversals per inch,* pentru șir continuu de tranziții.

Timp de acces la date, după poziționare: durata necesară pentru ca informația utilă pe discul în rotație să ajungă în dreptul capului poziționat.

Poziționare: stare pregătită a capului magnetic, „încărcat" (vezi § 2.1.2) pe suprafața de înregistrare în dreptul pistei dorite, gata de a efectua transferul de date.

Timp de poziționare pe pistă: interval de timp necesar pentru poziționarea capului magnetic pe o altă pistă dorită. Se compune din timpul de deplasare de la o pistă la alta și timpul de amortizare a oscilațiilor mecanice, pe pista dorită.

Timp maxim de poziționare: timp de poziționare pe pista 76 (0), pornind de la pista 0 (76).

Timp mediu de poziționare: timp de poziționare calculat pentru accesul la o pistă oarecare pornind de la o pistă oarecare (suma duratelor tuturor deplasărilor posibile împărțită la numărul de posibilități).

Timp de acces: timp de poziționare plus timp de acces la date.

Timp mediu de acces statistic: timp mediu de acces calculat pentru o operație complexă cu discul, caracterizat de raportul între timpul total de lucru și numărul total de accese, scăzînd timpul în care nu se lucrează cu discul.

Index: semnal care precizează un punct de referință pe circumferința pistei; apare în timpul rotației dischetei, la trecerea unui orificiu din mediul magnetic prin dreptul unei fante circulare, prevăzute în plic.

Pistă: spațiul de înregistrare accesibil capului magnetic la o rotație completă. Magnetizarea se face în direcția de parcurgere a pistei, prin saturație, în cele două sensuri [17].

Sector: unitate de înregistrare conținînd un număr standardizat de octeți. Este precedat de octetul (octeții)-marcă, în funcție de care se stabilește natura lui: normal sau special. Sectoarele speciale, defecte sau protejate, pot fi sărite la examinarea unei piste.

Decalaj de vîrf, peak-shift: variația poziției vîrfului impulsului citit în raport cu poziția sa normală. Se măsoară relativ la poziția impulsului precedent sau la o localizare medie a impulsurilor precedente, stabilită cu ajutorul unei bucle cu calare pe fază, PLL.

Formatare: operație inițială de premarcare; se înregistrează complet pista, cu conținut stabilit în sectoarele de date, ordinea numerelor de sectoare, înscrise în blocul de identificare, BI, poate fi normală (1 ÷ 1AH) sau aleatoare, conformă cu un tabel specificat.

2.1.2. INTERFAȚA MEDIU — CAP MAGNETIC

Interfața mediu — cap magnetic, prin care se efectuează transferul de informație, se realizează prin contact direct. Discheta, antrenată de un motor electric de curent alternativ sau de curent continuu (la ultimele tipuri), se află în mișcare de rotație cu viteză constantă. Capul magnetic este transportat pe un car, a cărui poziție față de o pistă de referință corespunde pistei cu care se face transferul. Capul se așază pe pistă prin apăsarea dischetei

între cap, pe de o parte, și un tampon cu pîslă, pe de altă parte, acesta din urmă fixat pe un braț acționat de un electromagnet. Odată ajuns la pista aleasă, capul magnetic rămîne fixat în timpul „încărcării", iar tamponul de pîslă aduce suprafața de înregistrare în apropierea întrefierului de înregistrare/redare. Capul magnetic creează o mică adîncitură pe suprafața dischetei, deformare care dispare după îndepărtarea de zona prinsă între cap și tampon. Suportul și capul magnetic sînt proiectate să reziste la frecarea intensă din zona transferului. Variațiile poziției capului, ale parametrilor mișcării carului și ale poziției dispozitivului de prindere a dischetei, stabilesc acuratețea poziționării pe pistă și excentricitatea parcurgerii ei la diferite treceri, limitînd în bună măsură performanțele UDF.

2.1.3. DISCHETA

Suportul magnetic este protejat de un plic de formă pătrată, în care sînt prevăzute mai multe decupări, ca în figura 2.3:

— o decupare radială pentru accesul capului magnetic la fiecare pistă;

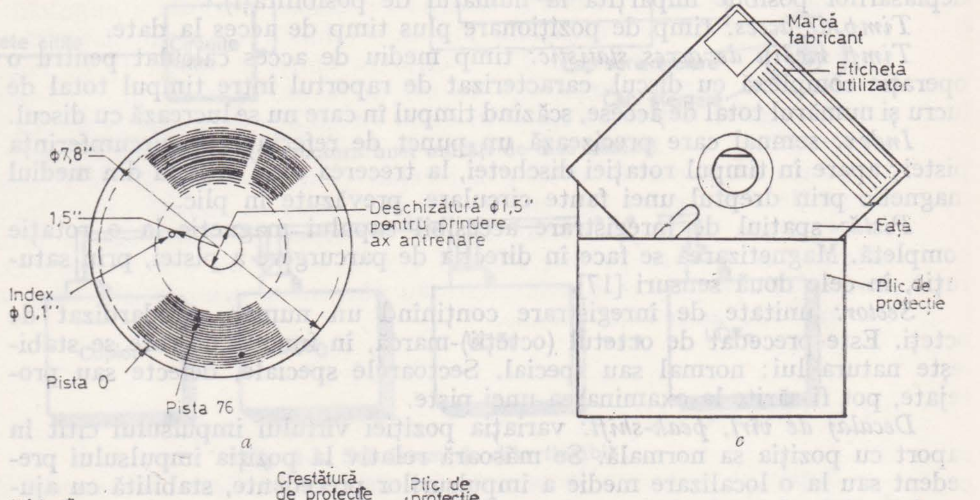


Fig. 2.3. Discheta de 8 *inch*:
a. suportul magnetic; b. decupări în dischetă; c. discheta în plicul de protecție; d. introducerea dischetei în UDF.

— o decupare circulară centrală, pentru prinderea discului în axul de antrenare al UDF;

— o mică decupare circulară excentrică, în apropierea decupării centrale, pentru sesizarea trecerii perforației corespunzătoare din suportul magnetic, în vederea generării semnalului INDEX, destinat detectării stării READY și stabilirii unei poziții de referință pentru fiecare pistă;

— o crestătură laterală, în afara zonei discului magnetic, utilizată pentru protecția la scriere.

Caracteristicile mecanice și magnetice ale dischetei sînt standardizate pentru dischetele cu latura de 8 *inch* și de 5 $^1/_4$ *inch* (minidisc). Anumite variante pot fi adoptate pentru înregistrarea pe o față sau pe ambele, pentru UDF cu unul/două capete, pentru înregistrări în densitate simplă/dublă.

2.1.4. CARACTERISTICI ALE UNITĂȚILOR DE DISC FLEXIBIL

Se dau mai jos caracteristicile tehnice ale unei UDF tipice pentru dischete de 8 *inch:* unitatea CDC 9404, cu antrenare cu șurub, inclusă în echipamentul UDF 101, produs la IEPER-București.

2.1.4.1. CARACTERISTICI GENERALE

Diametru disc flexibil (suport magnetic):	7,88 *inch* (20,02 cm)
Latură dischetă:	8 *inch* (20,32 cm)
Viteză de rotație ($\pm$ 3,5%):	360 rot./min (6 rot./s)
Raza pistei (N=0÷76):	$2{,}029 + \frac{76 - N}{48}$ *(inch)*
	$51{,}537 + 25{,}4\frac{76 - N}{48}$ (mm)

Notă: Variația vitezei de rotație cu $\pm$3,5% include și variațiile datorate unei deviații a frecvenței de curent alternativ cu $\pm$2% ($\pm$ 1 Hz la frecvența europeană de 50 Hz).

2.1.4.2. CARACTERISTICI ALE ÎNREGISTRĂRII

Mod de înregistrare:	Densitate simplă, FM	Densitate dublă, MFM
Densitate înregistrare pistă exterioară (0):	1 836 bpi (72,3 bit/mm)	3 672 bpi (144,6 bit/mm)
Densitate înregistrare pistă interioară (76):	3 257 bpi (128,7 bit/mm)	6 536 bpi (257,3 bit/mm)
Viteză transfer:	250 Kbit/s:	500 Kbit/s
Celulă-bit:	4 μs	2 μs
Octeți/pistă (neformatat):	5 208 (≈ 5 Kocteți)	10 416 (≈ 10 Kocteți)
Octeți/dischetă (neformatat, o față):	401 016 (≈ 400 Kocteți)	802 032 (≈ 800 Kocteți)
Număr de piste:	77	77

Număr de sectoare/pistă:	Format standard (26, 15, 8, 4)	
Capacitate pistă (format standard 26 sectoare):	3 328 (3 $^1/_4$ Kocteţi)	(6 656 6 $^1/_2$ Kocteţi)
Capacitate dischetă (format standard 26 sectoare, o faţă):	256 256 (256 Kocteţi)	512 512 (0,5 Mocteţi)

2.1.4.3. TIMPI DE ACCES

Timpi de acces la date (după poziţionare):

— timp mediu nominal (1/2 tură, la 6 rot./s) = 83,33 ms;
— timp maxim (1 tură, la 6 rot./s+3,5%) = 173 ms.

Timpi de poziţionare pe pistă:

— timp de acces la pista adiacentă, *Step Rate*, SR = 10 ms;
— timp de amortizare oscilaţii mecanice la acces pistă, *Head Settling Time*, HST = 10 ms;
— timp de încărcare cap magnetic, *Head Load Time*, HLT = 60 ms;
— timp mediu de poziţionare pe o pistă oarecare = 267 ms;
— timp maxim de poziţionare 770 ms.

2.1.4.4. CARACTERISTICI REFERITOARE LA CAPUL MAGNETIC

Lăţime pistă:		0,013 *inch* (0,33 mm).
Distanţa între axele pistelor:		0,02083 *inch* (0,529 mm).
Densitate de piste:		48 piste/mm
Lăţime cap ştergere:		0,035 *inch* (0,889 mm).
Decalaje de vîrf, FM:	referinţă impuls precedent	± 1 μs,
	referinţă PLL	± 0,5 μs,
MFM:	referinţă impuls precedent	± 0,5 μs,
	referinţă PLL	± 0,35 μs.

2.1.4.5. CARACTERISTICI SPECIFICE UDF DE TIP MF 6400

Pentru unităţi duale echipate cu UDF tip MF 6400, fabricate de firma MOM-Budapesta, caracteristicile generale şi ale înregistrării sînt identice cu cele ale tipului CDC 9404. Deoarece sistemul de antrenare al carului ce poartă capul magnetic foloseşte un procedeu avansat pentru traducerea mişcării de rotaţie a motorului pas cu pas în mişcare de translaţie a capului magnetic de la o pistă la alta (s-a înlocuit antrenarea prin contact direct, cu sanie şi şurub, printr-un sistem cu sanie şi bandă elastică), timpii de acces la pistă s-au micşorat, iar fiabilitatea a crescut. La acest tip de unităţi, timpii de poziţionare pe pistă sînt:

— timp de acces la pistă adiacentă 4 ms;
— timp de amortizare oscilaţii macănice, HST 20 ms;

— timp de încărcare cap magnetic, HLT 45 ms;
— timp mediu de poziționare pe o pistă oarecare 122 ms;
— timp maxim de poziționare 324 ms.

Timpii de acces la date (sector) nu depind de tehnologie și se păstrează ca la CDC 9404.

2.1.5. SEMNALE DE INTERFAȚĂ A UNITĂȚILOR DE DISC FLEXIBIL

Liniile de interfață sînt, de regulă, de tip cu colector-în-gol, active pe „0". La un singur FRM pot fi conectate, printr-un cablu care trece de la o unitate la alta, în lanț, *daisy chain*, mai multe (2÷4) UDF-uri, ca în figura 2.2. Selectarea se face prin semnale separate, recunoscute cîte unul singur în fiecare UDF. MOM 6400 are o opțiune cu recunoaștere a adresei codate binar pe 3 linii, ceea ce permite conectarea în lanț a maximum 8 UDF.

Circuitele electronice aferente UDF se compun din:
— circuite de comenzi și stare;
— circuite de scriere;
— circuite de citire;
— surse de alimentare (eventual comune la mai multe UDF).

Circuitele de scriere/citire realizează transferul de informații între interfața electrică a UDF (nivel logic) și capul magnetic (nivel analogic). Interfața mediu — cap magnetic transferă informația între nivelul electric analogic (prezență și direcție curent în înfășurări) și nivelul magnetic remanent (zone succesive cu magnetizare de sens opus). Circuitele de scriere transformă un șir de impulsuri în schimbări ale curentului de scriere, care conduc la tranziții între stările de magnetizare de pe suportul magnetic. Circuitele de citire prelucrează curentul indus în înfășurarea de citire a capului magnetic, corespunzător fiecărei tranziții în magnetizare, furnizînd impulsuri care se trimit pe interfața cu FRM.

Blocul de comandă și stare furnizează succesiunea de comenzi, pentru diferite subansamble ale UDF, în vederea stabilirii condițiilor normale de lucru, respectînd semnalele de interfață primite și emițînd, la rîndul său, semnale de stare, în conformitate cu traductoarele existente pe unitate. Se furnizează comenzi pentru acționarea motorului pas-cu-pas, după analiza semnalelor STEP și DIRECTION, se acționează solenoidul de încărcare a capului, conform comenzii HEAD LOAD, se trimit pe interfață semnalările LED-ului de INDEX, ale LED-ului de WRITE PROTECT și se reunesc mai multe condiții pentru generarea semnalelor READY (unitate pregătită pentru transfer) și WRITE FAULT. Semnalul WRITE FAULT se memorează într-un bistabil, astfel că pentru începerea unei alte operații de scriere, acesta trebuie șters cu un semnal de interfață WRITE FAULT RESET.

Dăm în continuare o descriere detaliată a semnificației semnalelor de interfață, pe baza cărora se construiește FRM, pentru CDC 9404. Între paranteze se dau eventualele diferențe, pentru MOM 6400. Pe interfață se folosesc semnalele active pe „0", astfel ca în absența FRM, intrările neconectate ale UDF să reprezinte comenzi inactive.

STEP, pas. Un impuls la „1" logic, de minimum 10 μs (1 μs) conduce la deplasarea capului pe o pistă alăturată spre interior sau exterior, în funcţie de semnalul DIRECTION, stabilit cu cel puţin 1 μs înainte. Impulsurile succesive se pot da la intervale de minimum 10 ms (4 ms). După ultimul impuls se produc oscilaţii mecanice, în jurul poziţiei de echilibru, care se amortizează în 20 ms.

DIRECTION, direcţie. Nivelul logic „1" determină direcţia de deplasare a capului magnetic spre interior. Nivelul „0" determină deplasarea spre exterior. Starea liniei trebuie fixată cu minimum 1 μs înaintea lansării comenzii STEP.

UNIT SELECT 1 ÷ 4, selecţie UDF. Nivelul logic „1" pe o singură linie selectează una dintre cele 4 unităţi, în vederea transferului cu FRM. (La MOM 6400 se pot conecta pînă la 8 UDF, prin codarea binară a adresei).

UNIT READY 1 ÷ 4, UDF pregătită. Nivelul logic „1" arată că unitatea respectivă este pregătită de lucru şi că semnalele de INDEX sosesc într-un interval stabilit. (La unităţile MOM 6400 există o singură linie READY, cu semnal corespunzător unităţii selectate). La CDC 9404 semnalele READY 1 ÷ 4 conţin starea unităţilor respective indiferent de selectare.

HEAD LOAD, încărcare cap. Un nivel logic „1" încarcă discul pe capul magnetic folosind tamponul presor de pe cealaltă parte a dischetei. În urma încărcării se produc oscilaţii mecanice care se amortizează în 60 ms (35 ms), după care se pot începe operaţiile de scriere sau citire. Pentru a preveni uzura inutilă a capului şi a mediului, se cere ca acest semnal să fie inactiv dacă nu se intenţionează o operaţie de transfer de date.

WRITE ENABLE, validare scriere. Un nivel logic „1" validează ca datele sosite pe linia WRITE DATA să fie înregistrate pe disc. Nivelul logic „0" validează ca datele înregistrate pe disc să fie trimise pe linia READ DATA.

WRITE DATA, date scriere. Un impuls de nivel logic „1", avînd durata de 250 ns ± 50 ns, conduce, pe frontul anterior, la o tranziţie în magnetizarea de pe suport. În FM, impulsurile sînt distanţate la 2 sau 4 μs, pentru o celulă-bit de 4 μs, în MFM la 2, 3 sau 4 μs, pentru o celulă-bit de 2 μs, iar în M^2FM* la 2, 3, 4 sau 5 μs, pentru aceeaşi celulă-bit. În densitate dublă se poate folosi, pentru pistele interioare, precompensarea cu decalaje de 150 ÷ ÷ 250 ns în raport cu poziţia de referinţă (vezi § 3.4.3.2).

INDEX. Un impuls „1" logic semnalează prezenţa perforaţiei din mediul magnetic în dreptul orificiului din plicul protector. Pentru discurile „sectorizate *soft*" se obţine o informaţie asupra poziţiei relative a discului faţă de unitate. Impulsul are lăţimea de 1,8 ms. Impulsurile succesive apar la 166,7ms, pentru durata unei rotaţii la viteza de 6 rot./s. Pentru dischete cu „sectorizare *hard*", care au încă 32 de perforaţii echidistante pe circumferinţa perforaţiei de INDEX, linia furnizează indicaţii asupra începutului sectoarelor de date. Prin modificări în UDF cu *strap*-uri, o dischetă cu „sectorizare *hard*" poate fi considerată, la interfaţa UDF, ca dischetă cu „sectorizare *soft*" — prezentînd un singur impuls de INDEX la o rotaţie.

---

* M^2FM, *Modified MFM*, MFM modificat: mod de înregistrare derivat din MFM, în care simbolul de sincronizare dintr-un cuvînt de cod lipseşte şi dacă există simbol de sincronizare în cuvîntul precedent.

TRACK 00, pista 0. Nivelul logic „1" indică prezența capului magnetic în dreptul pistei cu circumferința maximă, numită și pistă de referință.

READ DATA, date citite. Impulsuri, nivelul logic „1" reflectă prezența, în mediul ce trece prin fața capului magnetic, a tranzițiilor în magnetizare. Duratele corespund celor de la WRITE DATA, iar localizarea impulsurilor prezintă variații provocate de decalajul de vîrf. CDC 9404 asigură decalaj de vîrf între $\pm$ 0,5 μs (FM) și între $\pm$ 0,35 μs (MFM) pentru cazul unei măsurări avînd referința mediată cu buclă PLL.

WRITE PROTECT, protecție la scriere. Nivel logic „1" semnalează prezența crestăturii de pe marginea dischetei. Operațiunile de scriere sînt invalidate, chiar dacă se emit comenzi în acest sens. Nu se acționează bistabilul WRITE FAULT.

WRITE FAULT, eroare în scriere. Semnal prezent numai la CDC 9404 care, pe „1" logic, semnalează apariția unei situații anormale în timpul operației de scriere, impuse de activarea liniei WRITE ENABLE.

WRITE FAULT RESET, inițializare memorare eroare scriere. La „1" logic se șterge bistabilul din UDF care a memorat apariția unei stări de eroare la scriere, WRITE FAULT (numai la CDC 9404).

LOW CURRENT, comandă curent scriere. Semnal care, la „1" logic, schimbă valoarea curentului de scriere. Se folosește pentru a micșora interferența intersimbol la densitatea de înregistrare mai mare de pe pistele interioare pistei 43.

2.2. FORMATE STANDARD PENTRU ÎNREGISTRARE PE DISC FLEXIBIL

2.2.1. FORMAT STANDARD PENTRU ÎNREGISTRARE ÎN DENSITATE SIMPLĂ

Pentru densitate simplă, în mod de înregistrare FM, cel mai răspîndit format corespunde soluției adoptate în sistemul IBM 3740, devenit standard ISO 5654.

Înregistrarea în mod FM corespunde unui raport de codare $^{1}/_{2}$. Cel două simboluri de cod ale unei celule-bit au următoarele semnificații: primul simbol constituie impuls de „ceas" pentru sincronizare, iar al doilea corespunde informației ce se înregistrează. Tabelul de codare va fi:

Date	Simboluri de cod
1	11
0	10

Cuvintele 00 şi 10 fiind cuvinte invalide în codul FM, decodarea este instantanee, deoarece al doilea simbol al cuvîntului de cod conţine chiar informaţia de înregistrat, astfel că este suficientă o singură celulă-bit pentru refacerea datelor memorate.

Formatul unei piste înregistrate în standardul pentru densitate simplă, ca în figura 2.4, cuprinde 26 de sectoare, fiecare avînd 128 octeţi utili. Restul simbolurilor de pe pistă sînt redundante, folosindu-se la localizarea, sincronizarea şi verificarea datelor utile. Cele 26 de sectoare sînt incluse în 26 de *zone* cu structură identică, separate prin intervale de 33 octeţi. Între ultima

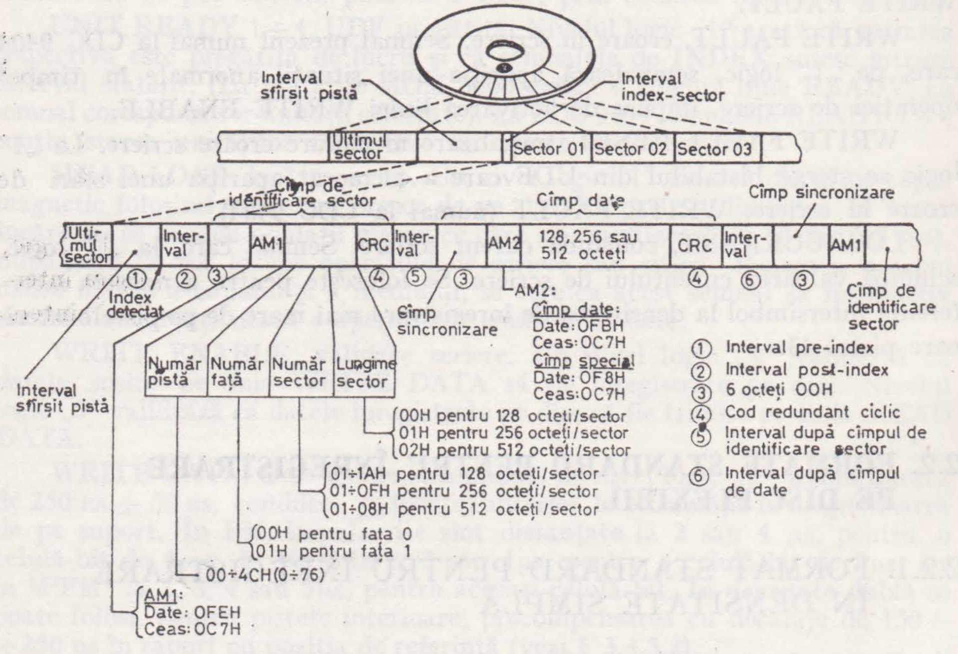


Fig. 2.4. Formatul standard pentru înregistrare în densitate simplă

zonă şi prima se află un interval ce conţine un număr variabil de octeţi. În funcţie de variaţiile de viteză ale discului, acest număr este în jur de 350 de octeţi. O zonă cuprinde un bloc de identificare, BI, un interval de sincronizare şi sectorul de date. BI conţine informaţii privind numerele de ordine al sectorului, al pistei, al feţei şi dimensiunile sectorului. Atît BI cît şi sectorul încep prin cîte un octet special, numit *marcă*, în care nu se mai respectă regulile de codare ale modului de înregistrare FM. *Violarea de cod,* care constă în absenţa unor simboluri de sincronizare din unele celule-bit ale mărcii, devine referinţă pentru stabilirea începutului de octet la blocul care urmează. O marcă specială se prevede în intervalul între ultima zonă şi prima zonă de pe pistă, pentru semnalarea zonei de INDEX. Tipurile de mărci existente în formatul de înregistrare în densitate simplă se dau în tabelul 2.1.

TABELUL 2.1. **Tipurile de mărci în format de densitate simplă.**

Celule-bit	7	6	5	4	3	2	1	0	Date/ceas. Hexazecimal
Marca de index, MIX	1 1	1 1	⓪ 1	1 1	⓪ 1	1 1	1 0	1 0	0FCH/0D7H
Marca de bloc de identificare, MBI	1 1	1 1	⓪ 1	⓪ 1	⓪ 1	1 1	1 1	1 0	0FEH/0C7H
Marca de sector normal, MSN	1 1	1 1	⓪ 1	⓪ 1	⓪ 1	1 0	1 1	1 1	0FBH/0C7H
Marca de sector special, MSS	1 1	1 1	⓪ 1	⓪ 1	⓪ 1	1 0	1 0	1 0	0F8H/0C7H

Note: 1. Înregistrarea simbolurilor de cod FM se face de la stînga spre dreapta.
2. ⓪ corespunde unui impuls de sincronizare absent și constituie violarea de cod **FM**.

Fiecare BI și sectorul corespunzător sînt prevăzute cu cîte doi octeți terminali, pentru detecția de eroare. Controlul erorii se face folosind cod ciclic detector de eroare, CRC, aplicat numai asupra simbolurilor de informație din centrul celulelor-bit. În calcul intră și mărcile de început de bloc. Cei 16 biți de control ai ultimilor 2 octeți se calculează cu un registru CRC, astfel ca polinomul blocului, preînmulțit cu un anumit factor, să fie un multiplu al polinomului generator $x^{16} + x^{12} + x^5 + 1$. Poziția simbolului în bloc corespunde ponderii sale, ponderea descrescînd succesiv spre sfîrșitul blocului. Preînmulțirea se realizează prin poziționarea la „1" a tuturor bistabililor registrului CRC, înaintea introducerii informațiilor (vezi și § 3.4.).

Structura stabilită pentru formatul pistei permite accesul aleator la sector, prin identificarea BI propriu fiecăruia. La scrierea unui sector se înregistrează în prealabil 6 octeți de sincronizare 00H și se adaugă un octet 0FFH după ultimul octet de control ciclic. De observat că, după scrieri succesive de sectoare, la zonele de trecere din intervalele BI-sector și sector-BI, se creează ruperi între șirul de biți înregistrați la formatare și șirul de simboluri din regiunile de început și sfîrșit ale sectorului.

2.2.2. FORMAT STANDARD PENTRU ÎNREGISTRARE ÎN DENSITATE DUBLĂ

Pentru densitate dublă, în mod de înregistrare MFM, cel mai răspîndit format corespunde soluției adoptate în sistemul IBM/34, devenit standard ISO 7065.

Înregistrarea în mod MFM corespunde unui raport de codare $^1/_2$, în care cele două simboluri de cod ale unei celule-bit au aceleași semnificații ca la modul FM, cu observația că simbolurile de „ceas" se elimină dacă în celula precedentă sau în cea curentă există simbol de „date".

Tabelul de codare va fi:

Date celulă precedentă	Date celulă curentă	Simboluri de cod
X	1	01
1	0	00
0	0	10

Cuvîntul 11 este invalid și, din aceleași motive ca și la modul de înregistrare FM, decodarea este instantanee.

Formatul unei piste, înregistrate în standard MFM, pentru densitate dublă, figura 2.5, cuprinde 26 de sectoare, fiecare a cîte 256 octeți, restul simbolurilor de pe pistă avînd aceleași semnificații de la formatul standard FM. Cele 26 de zone sînt separate prin intervale de 36 de octeți. Între prima și ultima zonă sînt aproximativ 790 octeți. Formatarea începe de la impulsul de index și se încheie la întîlnirea aceluiași impuls, astfel că numărul de octeți ai ultimului interval depinde de viteza discului, în timp ce restul înregistrării depinde de frecvența de referință a scrierii, obținută cu precizie de la un oscilator cu cuarț. O zonă include un BI, un interval de 33 octeți și sectorul de date. BI și sectorul de date au un conținut cu aceleași semnificații ca și la formatul FM standard, cu același polinom de verificare cu cod ciclic detector de erori și aceeași procedură de inițializare. Marca de BI și cea de sector au fost schimbate pentru a se conforma codului MFM.

Violarea de cod constă, ca și la FM, în lipsa unui simbol de „ceas". Se folosesc două tipuri de octeți cu violare. Unul este folosit la marca de index (0C2H/014H), iar celălalt, pentru BI și sector (0A1H/00AH) (vezi tabelul 2.2).

TABELUL 2.2. **Tipurile de mărci în format de densitate dublă**

Mărci \ Celula-bit	7	6	5	4	3	2	1	0	Date/ceas Hexazecimal
Octet marcă index	0 1	0 1	0 0	1 0	(0) 0	1 0	0 1	0 0	0C2H/014H
Octet marcă BI/sector	0 1	0 0	0 1	0 0	1 0	(0) 0	1 0	0 1	0A1H/00AH

Un bloc începe cu trei octeți de marcă MFM, urmați de un octet corect MFM, corespunzător octetului de date din mărcile FM. Astfel, se obțin combinațiile din tabelul 2.3.

În calculul CRC intră și octeții din mărci. Inițializarea registrului CRC se face înaintea intrării primului bit de date al primului octet cu violare de cod.

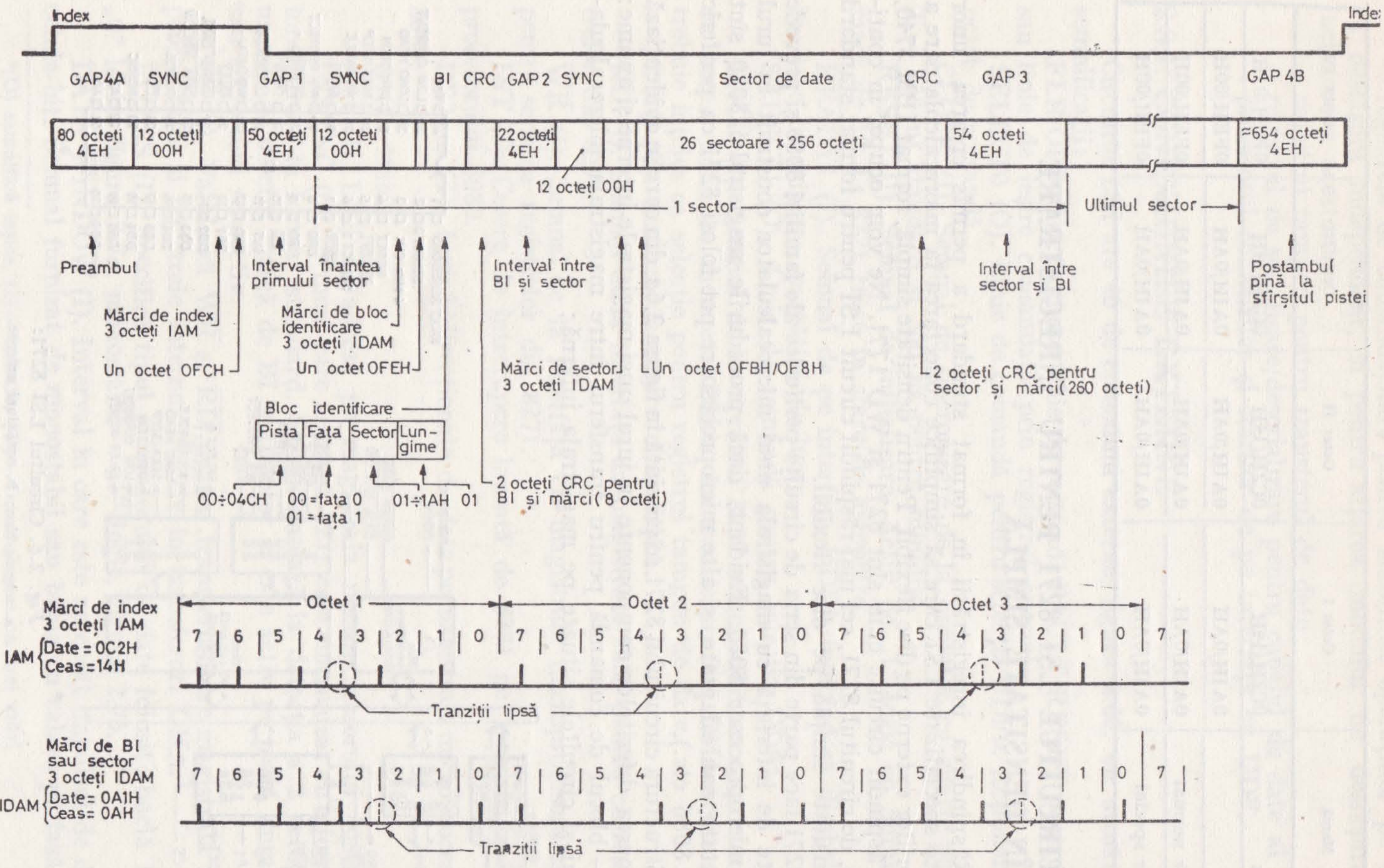


Fig. 2.5. Formatul standard pentru înregistrare în densitate dublă

TABELUL 2.3. **Combinații de mărci în format de densitate dublă**

Marcă	Octet I	Octet II	Octet III	Octet IV
Index	0C2H/14H	0C2H/14H	0C2H/14H	0FCH/01H
BI	0A1H/0AH	0A1H/0AH	0A1H/0AH	0FEH/00H
Sector normal	0A1H/0AH	0A1H/0AH	0A1H/0AH	0FEH/00H
Sector special	0A1H/0AH	0A1H/0AH	0A1H/0AH	0FBH/00H

2.3. CIRCUITUL LSI 8271 PENTRU ÎNREGISTRARE ÎN DENSITATE SIMPLĂ

Răspîndirea înregistrării în format standard a permis crearea unor circuite specializate LSI care să simplifice conectarea la microcalculatoare a memoriilor externe pe disc flexibil. Pentru densitate simplă, format IBM 3740, s-au răspîndit circuite cum sînt 8271 și WD 1771. Ne vom ocupa, în continuare, de circuitul 8271, cel mai răspîndit circuit LSI pentru format standard în densitate simplă pe disc flexibil.

8271 face parte din seria de circuite periferice ale familiei 8080 și prevede circuite de interfață cu magistrala unui microcalculator construit în jurul unui microprocesor 8080. Evident, dacă procedurile magistralei 8080 sînt respectate, sau adaptate, și alte microprocesoare pot folosi 8271 ca periferic (Z80, 8086 etc.).

Structura circuitului 8271 se prezintă în figura 2.6*a*, din care se evidențiază dispunerea diferitelor componente în jurul unei magistrale interne și anume:

— blocul de comandă pentru transferul între magistrala microcalculatorului și circuitele situate pe magistrala internă;

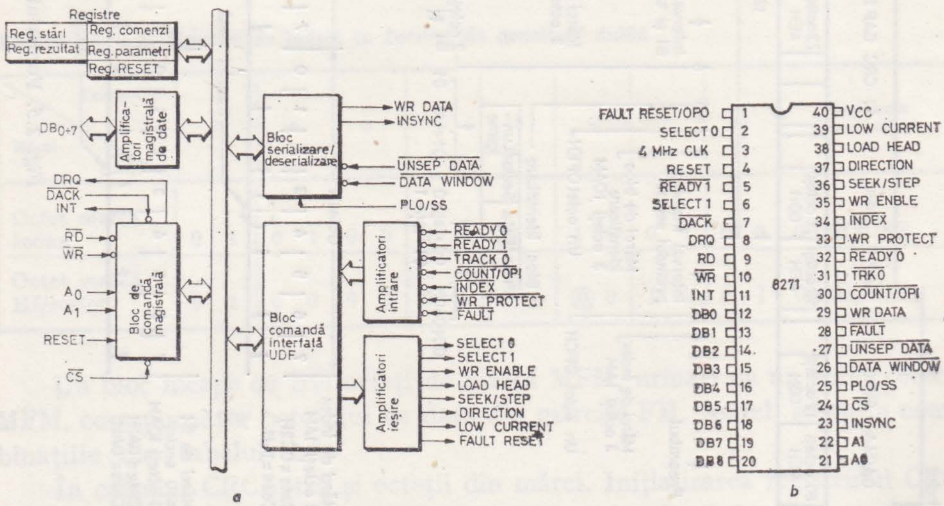


Fig. 2.6. Circuitul LSI 8271:
a. schema-bloc; b. conexiuni externe.

— registrele de comandă/stări pentru stabilirea operațiilor pe care le va desfășura independent, și pentru aflarea modului de desfășurare și încheiere a acestora;
— registrul tampon pentru transferul de date;
— registrul de serializare/deserializare pentru canalul de date al UDF;
— procesorul comenzilor și stărilor de pe interfața cu UDF.

2.3.1. CONEXIUNILE EXTERNE

Circuitul LSI are 40 de conexiuni externe, figura 2.6*b*, cu următoarele semnificații:

FLR/OPO (O)*. Opțiune: linie de comandă FAULT RESET pentru UDF sau linie de ieșire comandată prin registrul special.

SELECT0 (O). Linie de comandă pentru selecția UDF0. Acționare prin octet de comandă.

CLOCK (I). Semnal de sincronizare de la oscilator cu cuarț, perioada T = 250 ns standard și 500 ns pentru minidisc.

RESET (I). Semnal de inițializare. Activ pe „1" logic cu durată mai mare de 10 T; 8271 trece în stare de așteptare.

$\overline{\text{READY1}}$ (I). Semnal de pe interfața UDF1, indicînd că discheta este în stare de mișcare, pregătită de lucru.

SELECT1 (O). Linie de comandă pentru selecția UDF1.

$\overline{\text{DACK}}$ (I). Semnal de la circuitul 8257 semnificînd acceptarea ciclului de acces direct la memorie, DMA. La transferuri fără DMA acest semnal trebuie înțeles ca selecție pentru registrul tampon de date.

DRQ (O). Semnal de cerere ciclu DMA.

$\overline{\text{RD}}$ (I). Semnal de la interfața cu magistrala 8080 pentru citirea registrelor din 8271.

$\overline{\text{WR}}$ (I). Semnal de la interfața cu magistrala microprocesorului 8080 pentru scrierea registrelor din 8271.

INT (I). Cerere de întrerupere lansată de 8271 pe magistrala microprocesorului 8080.

DB7 ÷ 0 (T). Linii bidirecționale de date pe magistrala microprocesorului 8080.

GND. Masă.

A0,1 (I). Linii de adresă pe magistrala microprocesorului 8080.

INSYNC (O). Semnal care arată că 8271 este sincronizat cu informațiile înregistrate în format standard. Se declanșează la detecția a 2 octeți 00H urmați de octetul marcă de BI sau sector și este activ pe toată durata BI sau sectorului respectiv.

$\overline{\text{CS}}$ (I). Validează $\overline{\text{WR}}$ și $\overline{\text{RD}}$ pentru transferul între magistrala microprocesorului și registrele interne. Inactiv în transferuri DMA.

$\overline{\text{PLO}}$/SS (I). Specifică tipul separatorului de date folosit: buclă PLL— „0" logic, figura 2.7*a*, monostabil — „1" logic, figura 2.7*b*.

$\overline{\text{DATA WINDOW}}$ (I). Interval în care este așteptat impulsul de date în celula-bit. Semnal furnizat de monostabil sau bucla cu calare pe fază PLL.

*(O) semnifică ieșire, (I) — intrare, (T) — intrare/ieșire cu trei stări

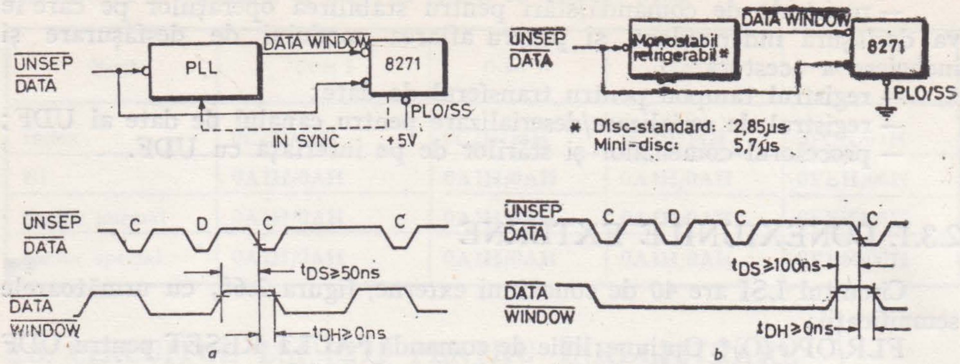
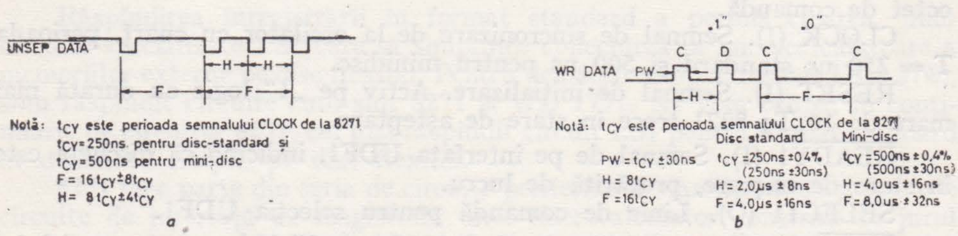


Fig. 2.7. Generarea ferestrei de date:
a. cu PLL; b. cu monostabil.



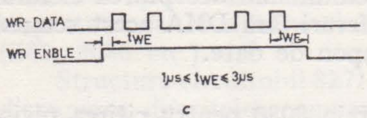


Fig. 2.8. Linii de date:
a. citire; b. scriere; c. validare scriere

UNSEP DATA (I). Semnal ce poartă impulsurile de citire cu caracteristicile din figura 2.8*a*.

FAULT (I). Semnal de la UDF care corespunde liniei WRITE FAULT.

WR DATA (O). Semnal cu caracteristicile din figura 2.8*b*, care corespunde liniei WRITE DATA de pe interfața cu UDF.

COUNT/OPI (O). Opțiune: linie utilizată pentru UDF care memorează numărul necesar de pași pînă la atingerea pistei dorite sau linie de intrare comandată prin registru special. În primul caz, prezentat în figura 2.9,

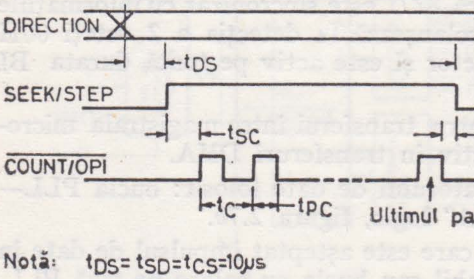
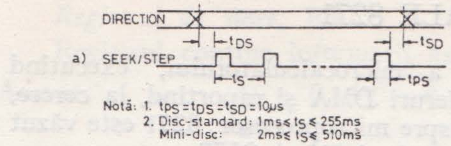


Fig. 2.9. Utilizarea liniei COUNT/OPI pentru căutare continuă



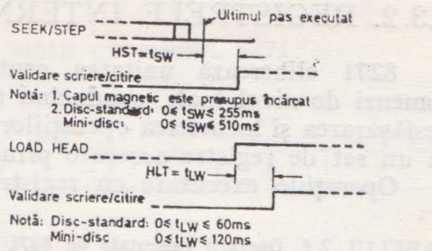


Fig. 2.10. Liniile pentru comanda carului:

a. SEEK/STEP și DIRECTION; b. întîrzierea între ultimul pas și validarea scrierii/citirii, HST; c. întîrzierea între comanda de încărcare și validarea scrierii/citirii, HLT

8271 este informat printr-un impuls la trecerea capului peste fiecare pistă.

$\overline{\text{TRK0}}$ (I). Semnal de la UDF care corespunde liniei TRACK 00.

$\overline{\text{READY0}}$ (I). La fel ca $\overline{\text{READY1}}$, dar pentru UDF0.

$\overline{\text{WR PROTECT}}$ (I). Semnal de la UDF, care corespunde liniei $\overline{\text{WRITE PROTECT}}$.

$\overline{\text{INDEX}}$ (I). Semnal de la UDF care corespunde liniei $\overline{\text{INDEX}}$.

$\overline{\text{WR ENABLE}}$ (O). Semnal spre UDF, care corespunde liniei $\overline{\text{WRITE ENABLE}}$. Validează linia $\overline{\text{WR DATA}}$, ca în figura 2.8*c*.

SEEK/STEP (O). Semnal spre UDF care corespunde liniei $\overline{\text{STEP}}$, figura 2.10*a*. Transferurile pot începe după o durată programabilă, ca în figura 2.10*b*.

DIRECTION (O). Semnal spre UDF care corespunde liniei $\overline{\text{DIRECTION}}$. Încadrează impulsurile de pe linia SEEK/STEP, figura 2.10*a*.

LOAD HEAD (O). Semnal spre UDF care corespunde liniei $\overline{\text{HEAD LOAD}}$. Transferurile pot începe după o durată programabilă, ca în figura 2.10*c*.

LOW CURRENT (O). Semnal spre UDF care corespunde liniei $\overline{\text{LOW CURRENT}}$.

VCC. Conexiune la care se aplică tensiunea de alimentare (+5 V).

Prezentăm, în figura 2.11, schema de principiu a conectării circuitului 8271 pe interfața cu UDF.

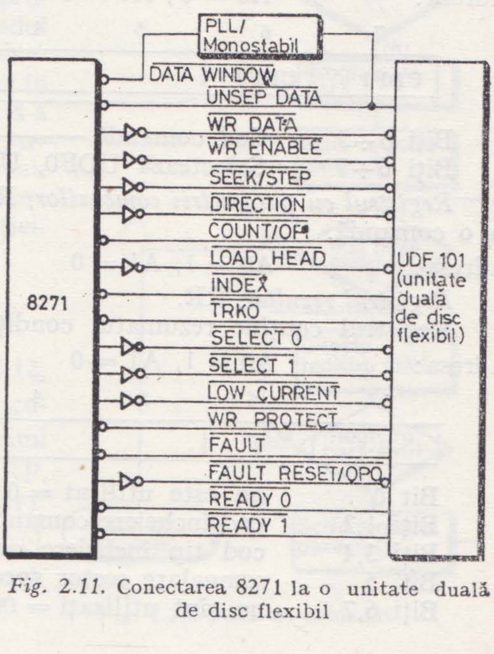


Fig. 2.11. Conectarea 8271 la o unitate duală de disc flexibil

2.3.2. REGISTRELE INTERNE ALE 8271

8271 eliberează unitatea centrală a microcalculatorului, executînd comenzi de nivel înalt, conducînd transferuri DMA şi raportînd, la cerere, desfăşurarea şi încheierea operaţiilor. Dinspre microprocesor, 8271 este văzut ca un set de registre accesate prin instrucţiuni IN şi OUT.

Operaţiile executate cu registrele 8271 sînt rezumate în tabelul 2.4.

TABELUL 2.4. **Operaţii executate de 8271**

Operaţie	$\overline{RD}$	$\overline{WR}$	A1	A0	$\overline{CS}$	$\overline{DACK}$
Citire stare	0	1	0	0	0	1
Scriere comandă	1	0	0	0	0	1
Citire rezultat	0	1	0	1	0	1
Scriere parametru	1	0	0	1	0	1
Reset	1	0	1	1	0	1
Scriere date	1	0	X	X	1	0
Citire date	0	1	X	X	1	0
Operaţie interzisă	X	X	X	X	0	0

Pentru aceste operaţii, 8271 dispune de următoarele registre de 8 biţi:

Registrul de comandă, RC.

Adresa: A0 = 0; A1 = 0

7	6	5	4	3	2	1	0
UDF1	UDF0						

Biţi 0÷5 Cod comandă

Biţi 6÷7 Selectează UDF0, UDF1.

Registrul cu parametrii comenzilor, RP. Pot fi ceruţi pînă la 5 parametri la o comandă.

Adresa: A0 = 1, A1 = 0

Registrul rezultat, RR.

Registrul conţine rezumatul condiţiilor de încheiere a unei comenzi.

Adresa: A0 = 1, A1 = 0

7	6	5	4	3	2	1	0
0	0						0

Bit 0 nu este utilizat = 0

Biţi 1,2 cod încheiere comandă

Biţi 3,4 cod tip încheiere comandă

Bit 5 semnalare sector special (marcă 0F8H/0C7H)

Biţi 6,7 nu sînt utilizaţi = 00

Registrul de stare, RS.

Registrul conţine informaţii asupra stării circuitului 8271.

Adresa: A0 = 0, A1 = 0

7	6	5	4	3	2	1	0
						0	0

Biţi 0,1	nu sînt utilizaţi, = 00
Bit 2	cerere de date fără DMA
Bit 3	cerere de întrerupere
Bit 4	RR plin
Bit 5	RP plin
Bit 6	RC plin
Bit 7	ocupat cu o comandă

Registrul reset, RRES.

Registrul permite iniţializarea circuitului 8271 prin program. Este activ dacă RRES conţine 01 pe o durată mai mare de 10T.

Buffer de date, BD.

Registru accesat prin program de către DMA ($\overline{DACK}$ = „0").

2.3.3. FAZELE DE LUCRU 8271

8271 lucrează în 3 faze succesive: faza de comandă, FC, faza de execuţie, FE şi faza de rezultat, FR. În FC, microprocesorul scrie în registrele RC şi RP codul şi parametrii comenzii. În FE, 8271 execută independent comanda primită, iar în FR, semnalează microprocesorului că s-a executat comanda, microprocesorul urmînd să testeze conţinutul unuia sau al mai multor registre pentru a determina modul de încheiere a operaţiei.

Faza de comandă FC

După organigrama din figura 2.12, dacă 8271 nu este ocupat cu altă comandă în curs de execuţi e, el poate primi comanda în RC şi 1÷5 parametri în RP. RP este organizat ca o memorie LIFO, *Last-In First-Out,* astfel că următorul parametru se poate înscrie după ce toată stiva a fost coborîtă cu un nivel.

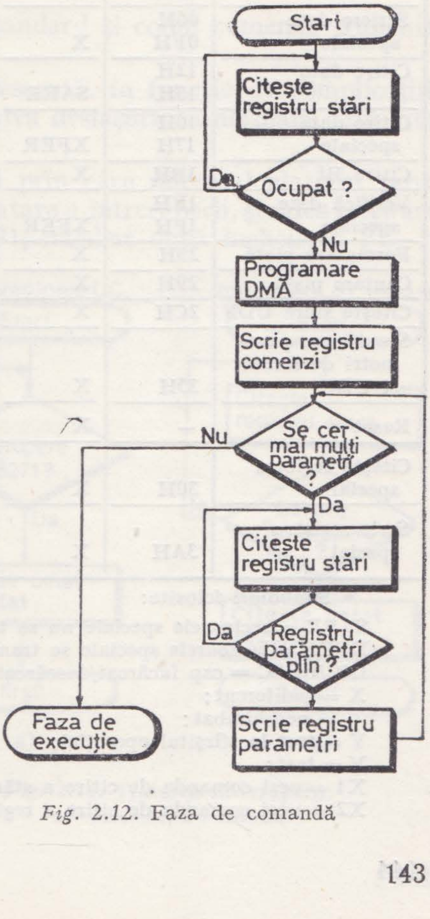


Fig. 2.12. Faza de comandă

Faza de execuție FE

8271 lucrează independent de microprocesor, realizînd comanda specificată în faza precedentă. Tabelul 2.5 rezumă unele dintre caracteristicile operațiilor realizate. Descrierea detaliată se va face în § 2.3.5., pentru fiecare comandă în parte. De remarcat următoarele:

— încărcarea capului se face după ce acesta a ajuns în dreptul pistei căutate;

— starea $\overline{READY} = 1$ (NOT READY) pentru o UDF selectată este memorată pînă la o comandă de citire stare UDF;

TABELUL 2.5. **Comenzile LSI 8271 ***

Comandă	Cod	Sectoare speciale	Încăr-care-descăr-care cap	Test READY	Test protecție scriere	Căutare pistă	Test pistă corectă	FR	Între-rupere
Căutare text (SCAN)	00H	SARE	INC	V	X	Da	Da	Da	Da
Căutare extinsă de text (SCAN)	04H	XFER	INC	V	X	Da	Da	Da	Da
Scriere date	0AH 0BH	X	INC	V	V	Da	Da	Da	Da
Scriere date speciale	06H 0FH	X	INC	V	V	Da	Da	Da	Da
Citire date	12H 13H	SARE	INC	V	X	Da	Da	Da	Da
Citire date speciale	16H 17H	XFER	INC	V	X	Da	Da	Da	Da
Citire BI	1BH	X	INC	V	X	Da	Nu	Da	Da
Verifică date speciale	1EH 1FH	XFER	INC	V	X	Da	Da	Da	Da
Formatare pistă	23H	X	INC	V	V	Da	Nu	Da	Da
Căutare pistă	29H	X	INC	Y	X	Da	Nu	Da	Da
Citește stare UDF	2CH	X	—	X	X	Nu	Nu	X1	Nu
Specificare parametri de format și UDF	35H	X	—	X	X	Nu	Nu	Nu	Nu
Reset	—	X	DESC	X	X	Nu	Nu	Nu	Nu
Citește registru special	30H	X	—	X	X	Nu	Nu	X2	Nu
Scrie registru special	3AH	X	—	X	X	Nu	Nu	Nu	Nu

* Simboluri folosite:

SARE = sectoarele speciale nu se transferă;
XFER = sectoarele speciale se transferă;
INC/DESC = cap încărcat/descărcat;
X = indiferent;
— = neschimbat;
Y = test la sfîrșitul operației;
V = test;
X1 = vezi comanda de citire a stării UDF;
X2 = vezi comanda de citire a registrului special.

— la o comandă de citire normală a unui număr de sectoare, dacă un sector căutat este special, vezi tabelul 2.1, acesta nu se transferă, dar contează ca depăşit şi se trece, dacă este cazul, la transferul sectorului următor;

— orice operaţie care presupune căutare, execută implicit căutarea, fără a mai fi necesară o comandă separată.

Faza de rezultat FR

8271 intră în această fază din FE după executarea corectă a comenzii sau după apariţia unei erori în timpul execuţiei. Conform organigramei din figura 2.13, se testează în prealabil tipul rezultatului, care poate fi:

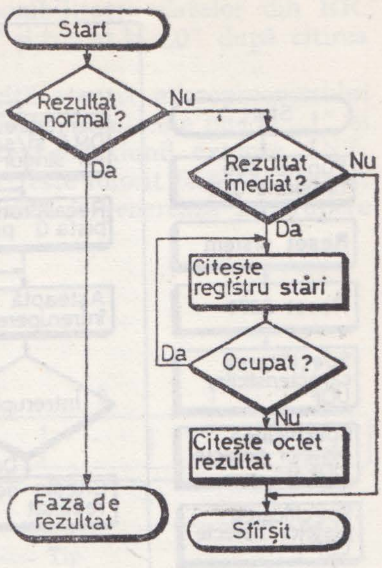


Fig. 2.13. Faza de rezultat

— standard, cu codul comenzii (biţii $0 \div 5$ ai RC) $\leqslant$ 02CH;

— rezultat imediat, dacă nu este standard şi codul comenzii trunchiat (biţii $0 \div 3$ ai RC) este $\geqslant$ 0CH.

FR poate avea o desfăşurare arborescentă, în funcţie de complicaţiile care pot apărea, cu o durată minimă pentru desfăşurarea normală, fără erori sau evenimente speciale.

Figura 2.14, prezintă cele două căi prin care microcalculatorul poate citi RR: calea hardware, prin logica de tratare a întreruperii, şi calea software, prin testarea bitului INT din RS al 8271, imediat după încheierea FC. În primul caz, pe durata FE, unitatea centrală devine disponibilă pentru alte operaţii.

Faza de iniţializare, FI

În afara celor 3 faze de funcţionare curentă, 8271 trebuie să treacă, imediat după resetarea sistemului, printr-o procedură de programare a modului propriu de lucru, faza de iniţializare. Aceasta trebuie să corespundă succesiunii de operaţii prezentate în organigrama din figura 2.15. 8271 este informat asupra parame-

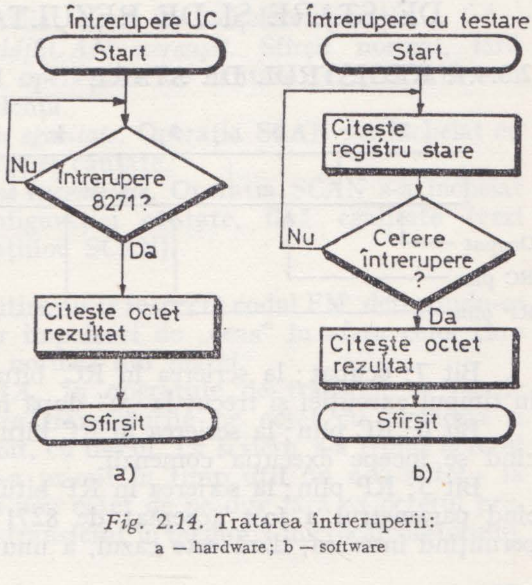


Fig. 2.14. Tratarea întreruperii:
a — hardware; b — software

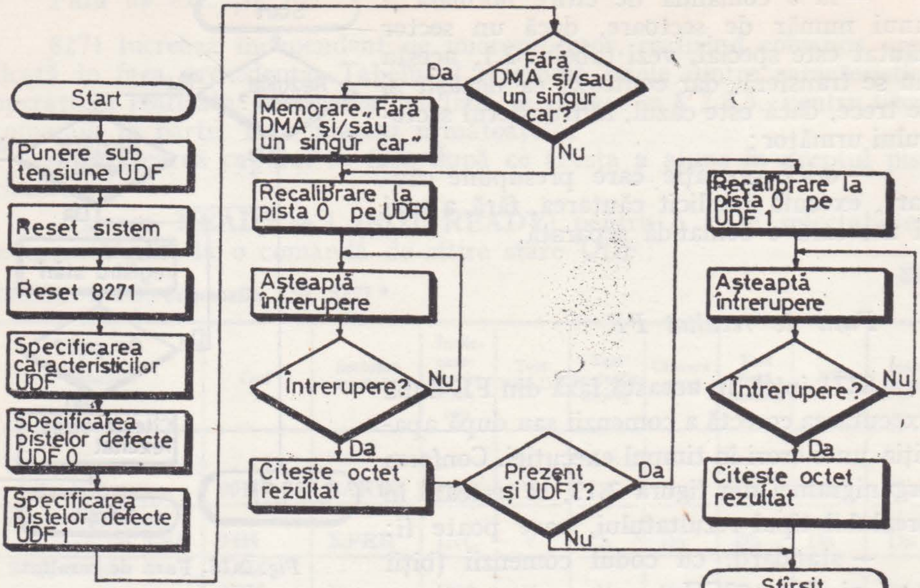
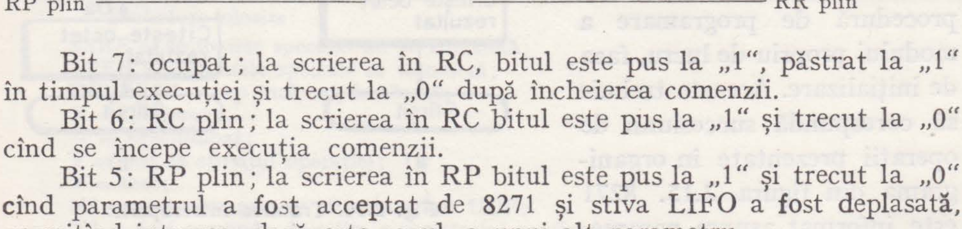


Fig. 2.15. Faza de inițializare

trilor UDF și stării curente a lor, ca și asupra configurației sistemului. FI include obligatoriu operații de retragere a carelor la poziția de referință pentru fiecare UDF.

2.3.4. DESCRIEREA REGISTRELOR DE STARE ȘI DE REZULTAT

2.3.4.1. REGISTRUL DE STARE



Bit 7: ocupat; la scrierea în RC, bitul este pus la „1", păstrat la „1" în timpul execuției și trecut la „0" după încheierea comenzii.

Bit 6: RC plin; la scrierea în RC bitul este pus la „1" și trecut la „0" cînd se începe execuția comenzii.

Bit 5: RP plin; la scrierea în RP bitul este pus la „1" și trecut la „0" cînd parametrul a fost acceptat de 8271 și stiva LIFO a fost deplasată, permițînd intrarea, dacă este cazul, a unui alt parametru.

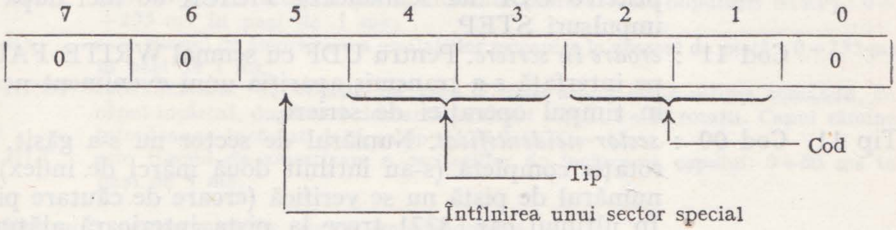
Bit 4: RR plin; bit care semnifică disponibilitatea datelor din RR. La încheierea unei comenzi bitul este pus la „1" şi trecut la „0" după citirea RR de microprocesor.

Bit 3: cerere întrerupere; dacă 8271 solicită atenţia microprocesorului la încheierea normală sau anormală a unei operaţii, bitul este pus la „1" şi trecut la „0" după citirea RR. Reflectă starea conexiunii externe INT.

Bit 2: cerere transfer fără DMA; dacă 8271 este folosit fără 8257, acest bit indică cereri de transfer date. În mod fără DMA se generează întrerupere pentru fiecare octet de date transferat.

Bit 1,0: Neutilizaţi (biţii la „0").

2.3.4.2. REGISTRUL DE REZULTAT



Biţi 7,6 : neutilizaţi (biţii la „0").
Bit 5 : s-a întîlnit cel puţin un sector special, cu marcă 0F8H/0C7H.
Biţi 4,3 : descriere tip încheiere operaţie.

- 00 — operaţie încheiată normal.
- 01 — operaţie încheiată cu eroare recuperabilă.
- 10 — operaţie încheiată cu eroare recuperabilă, dar necesitînd o intervenţie specială a microprocesorului.
- 11 — operaţie încheiată cu eroare de program sau de UDF defectă.

Biţi 2,1 : cod care descrie detaliat tipul de încheiere operaţie.

Tip 00 Cod 00 : *încheiere normală/SCAN nereuşit*. Sfîrşit normal, fără eroare; în cazul operaţiilor de căutare text, SCAN, nu s-a găsit coincidenţa.

Cod 01 : *SCAN reuşit, cu egalitate*. Operaţia SCAN s-a încheiat cu găsirea configuraţiei căutate.

Cod 10 : *SCAN reuşit, cu inegalitate*. Operaţia SCAN s-a încheiat cu găsirea configuraţiei căutate, fără egalitate (vezi descrierea operaţiilor SCAN).

Cod 11 : nefolosit.

Tip 01 Cod 00 : *eroare ceas*. La citire nu se respectă codul FM, detectîndu-se inexistenţa unor impulsuri de „ceas" în afara celor care lipsesc în mod normal din mărci.

Cod 01 : *eroare ritm DMA*. În operaţiile de transfer, se pierde cursivitatea între transferurile pe octet cu memoria şi cele seriale, pe bit, cu discul. La scriere, s-a trimis octetul pe disc şi nu s-a primit în timp util un nou octet; la citire a sosit un nou octet de pe disc fără ca octetul precedent să fi fost transferat în memoria microcalculatorului.

Cod 10 : *eroare CRC pe BI.* Citirea BI nu se verifică prin cod CRC. Restul calculat nu coincide cu cel înregistrat la formatare.

Cod 11 : *eroare CRC pe sector.* Citirea sectorului nu se verifică prin cod CRC. Restul calculat din datele citite nu coincide cu cel înscris. Datele, deşi s-au transferat, trebuie considerate nevalabile.

Tip 10 Cod 00 : *UDF nepregătită.* S-a găsit UDF în starea NOT READY. Acest cod este şters numai după o comandă de citire stare UDF.

Cod 01 : *protecţie scriere.* S-a lansat o comandă ce presupune înregistrare pe o dischetă protejată la scriere. Discheta nu este modificată.

Cod 10 : *nu se găseşte pista 0.* În timpul unei comenzi de căutare a pistei 0 UDF nu semnalează TRACK 00 nici după 255 impulsuri STEP.

Cod 11 : *eroare în scriere.* Pentru UDF cu semnal WRITE FAULT, pe interfaţă s-a transmis apariţia unui eveniment nedorit în timpul operaţiei de scriere.

Tip 11 Cod 00 : *sector neidentificat.* Numărul de sector nu s-a găsit, la o rotaţie completă (s-au întîlnit două mărci de index) sau numărul de pistă nu se verifică (eroare de căutare pistă). În ultimul caz, 8271 trece la pista interioară alăturată. Dacă nici aici nu se identifică pista, se mai încearcă odată pe încă o pistă interioară alăturată. Dacă nici pe această pistă nu se identifică sectorul, se poziţionează eroare de sector neidentificat.

Bit 0 : neutilizat („0").

2.3.5. DESCRIEREA OPERAŢIILOR

2.3.5.1. INIŢIALIZAREA

Operaţia se îndeplineşte prin înscrierea şi păstrarea în RRES timp de cel puţin 10T a unui octet 01H. Anularea comenzii se face prin înscrierea 00H. Acţiunea comenzii constă în:

— inactivarea liniilor de control UDF;
— încheierea forţată a eventualei operaţii în curs;
— ştergerea RS;
— trecerea 8271 în starea de aşteptare a unei comenzi.

2.3.5.2. SPECIFICAREA PARAMETRILOR UDF.

Cod „035H" în RC.

Comanda trebuie executată după RESET, înaintea oricărei alte comenzi şi se încheie după FC. În funcţie de primul parametru, PARAM 0, comanda poate fi:

— iniţializare (00H);
— specificare piste defecte UDF0 (10H);
— specificare piste defecte UDF1 (18H).

Format comandă specificare inițializare:

	7	6	5	4	3	2	1	0	
COMANDA	0	0	1	1	0	1	0	1	RC
PARAM 0	0	0	0	0	1	1	0	1	RP
PARAM 1	SR								RP
PARAM 2	HST								RP
PARAM 3	NROT				HLT				RP

unde: SR este timpul de acces la pista adiacentă (durata între impulsuri STEP) : $0\div$ $\div 255$ ms în pași de 1 ms;

RST este timpul de amortizare a oscilațiilor mecanice la accesul de pistă: $0\div 255$ ms în pași de 1 ms;

NROT este numărul de rotații pe care le face discul de la ultima comandă, cu capul încărcat, după care descarcă automat capul: $0\div 14$ rotații. Capul rămîne întotdeauna încărcat dacă se specifică NROT = 15.

HLT este timpul de amortizare a oscilațiilor de încărcarea capului: $0\div 60$ ms în pași de 4 ms.

Observație; peutru minidisc, timpii trebuie dublați.

Format comandă specificare piste defecte:

	7	6	5	4	3	2	1	0	
COMANDA	0	0	1	1	0	1	0	1	RC
PARAM 0	0	0	0	1	0/1	0	0	0	RP
PARAM 1	PRIMA PISTĂ DEFECTĂ								RP
PARAM 2	A DOUA PISTĂ DEFECTĂ								RP
PARAM 3	PISTA CURENTĂ								RP

Adresele pistelor defecte sînt specificate prin adresele lor fizice. Prezența unei piste defecte presupune că la pistele din interiorul acesteia, adresele fizice nu coincid cu adresele logice înscrise în BI. La inițializare se recomandă ca adresele de piste defecte și curentă să fie 0FFH.

2.3.5.3. CĂUTAREA

Cod „029H" în RC. Pentru UDF0: „069H" în RC, iar pentru UDF1: „0A9H" în RC.

Format:

	7	6	5	4	3	2	1	0	
COMANDA	UDF1	UDF0	1	0	1	0	0	1	RC
PARAM 0	PISTA FINALĂ ($0\div 255$)								RP

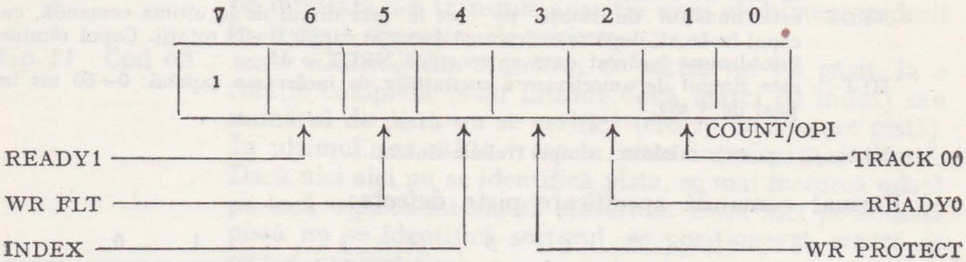
Comanda deplasează capul UDF specificat spre pista finală specificată. Capul nu se încarcă şi pista nu se verifică. Se folosesc indicaţiile preînregistrate referitoare la pistele defecte, pentru a realiza deplasarea la pista corectă.

Căutarea pistei 0, numită şi operaţie de *recalibrare*, se face pînă la recepţia semnalului TRACK 00. Dacă după 0FFH impulsuri de STEP nu se primeşte TRACK 00, se poziţionează un cod de eroare.

2.3.5.4. CITIREA STĂRII UDF

Cod „02CH", în RC. Pentru UDF0: „06CH" în RC, iar pentru UDF1: „0ACH" în RC.

În urma execuţiei se va obţine octetul rezultat în RR, care cuprinde starea conexiunilor externe ale circuitului 8271, corespunzătoare interfeţei cu UDF:



Trecerea prin READY = 0 se memorează. Dacă UDF a trecut prin NOT READY, se sesizează starea READY ulterioară prin două comenzi succesive de citire stare UDF : prima şterge memorarea NOT READY, a doua testează noua stare.

2.3.5.5. FORMATAREA

Cod „023H" în RC. Pentru UDF0 : „063H" în RC, iar pentru UDF1 : „0A3H" în RC.

Format:

	7	6	5	4	3	2	1	0	
COMANDA	UDF1	UDF0	1	0	0	0	1	1	RC
PARAM 0	ADRESĂ PISTĂ								RP
PARAM 1	GAP 3–6								RP
PARAM 2	LUNS				NRS/TK				RP
PARAM 3	GAP 5–6								RP
PARAM 4	GAP 1–6								RP

Comanda conduce la inițializarea pistei specificate la ADRESĂ PISTĂ. Sînt programabile lungimea sectoarelor LUNS, numărul de sectoare pe pistă NRS/TK, intervalele GAP 1 (marca index — primul BI), GAP 3 (sector — BI) și GAP 5 (semnal INDEX — marca INDEX).

În timpul operației, trebuie să fie furnizate cîmpurile BI pentru fiecare sector. 8271 generează mărcile și adaugă octeții CRC. Sectoarele sînt completate cu 0E5H. PARAM 1, 3 și 4 specifică lungimea șirului de octeți 0FFH din intervale. Înainte de BI și sector se scriu 6 octeți 00H pentru sincronizare. Intervalul BI — sector este fixat și conține întotdeauna 11 octeți 0FFH și 6 octeți 00H.

Tabelul 2.6 conține formatele posibile pentru disc standard de 8 *inch* și pentru disc standard de 5 $^1/_4$*inch*, minidisc.

2.3.5.6. CITIREA UNUI BLOC DE IDENTIFICARE

Cod „01BH" în RC. Pentru UDF0: „05BH" în RC, iar pentru UDF1: „09BH" în RC.

Format:

	7	6	5	4	3	2	1	0	
COMANDA	UDF1	UDF0	0	1	1	0	1	1	RC
PARAM 0	ADRESĂ PISTĂ								RP
PARAM 1	0	0	0	0	0	0	0	0	RP
PARAM 2	NUMĂR CÎMPURI BI								RP

Comanda transferă în memorie numărul de cîmpuri BI specificat în PARAM 2 de pe pista specificată cu PARAM 0, începînd cu primul BI după marca de INDEX. Din fiecare BI se transferă 4 octeți (numărul de pistă, numărul suprafeței, numărul de sector, lungimea sectorului) care se concatenează la șirul de octeți BI transferați anterior în ordinea așezării fizice pe disc.

2.3.5.7. CITIREA/SCRIEREA REGISTRELOR SPECIALE

Cod citire „03DH", iar cod scriere „03AH".

Citire registre speciale, RSP:

	7	6	5	4	3	2	1	0	
COMANDA	UDF1	UDF0	1	1	1	0	1	0	RC
PARAM 0	ADRESA RSP								RP

Rezultatul citirii se găsește în RR.

TABELUL 2.6. **Formate posibile conform standardelor de înregistrare în densitate simplă**

Număr de sectoare pe o pistă	INDEX – BI		BI		BI – Sector		Sector			Sector – BI		După ultimul sector		
	GAP 1				GAP 2					GAP 3		GAP 4	GAP 5	
	Marcă 0FCH + 0FFH	00H	Marcă 0FFH + Date	CRC	0FFH	00H	Marcă 0FBH sau marca 0F8H	Date	CRC	0FFH	00H	0FFH	0FFH	00H
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Pentru UDF de 8″:														
26	1+26	6	1+4	2	11	6	1	128	2	27	6	275	40	6
15	1+26	6	1+4	2	11	6	1	256	2	48	6	129	40	6
8	1+26	6	1+4	2	11	6	1	512	2	90	6	146	40	6
4	1+26	6	1+4	2	11	6	1	1 024	2	224	6	236	40	6
2	1+26	6	1+4	2	11	6	1	2 048	2	255	6	719	40	6
1	1+26	6	1+4	2	11	6	1	4 096	2	0	0	1 007	40	6
Pentru UDF de 5 1/4″:														
18	1+16	6	1+4	2	11	6	1	128	2	11	6	24	–	–
10	1+16	6	1+4	2	11	6	1	256	2	21	6	30	–	–
5	1+16	6	1+4	2	11	6	1	512	2	74	6	88	–	–
2	1+16	6	1+4	2	11	6	1	1 024	2	255	6	740	–	–
1	1+16	6	1+4	2	11	6	1	2 048	2	0	0	1 028	–	–

Notă: UDF de 8″ conțin în total câte 5 208 octeți pe fiecare pistă;

UDF de 5 1/4″ conțin în total câte 3 125 octeți pe fiecare pistă.

Scriere RSP:

	7	6	5	4	3	2	1	0	
COMANDA	UDF1	UDF0	1	1	1	1	0	1	RC
PARAM 0	ADRESĂ RSP								RP
PARAM 1	OCTET DATE								RP

Registrele speciale sînt selectate prin adresele:

06H Număr sector curent (SCAN)
14H Octetul cel mai semnificativ al numărului de octet la care a apărut coincidența în operații de SCAN.
13H Octetul cel mai puțin semnificativ al numărului de octet la care a apărut coincidența în operația de SCAN.
12H Pistă curentă UDF0.
1AH Pistă curentă UDF1.
17H Registru de mod.
23H *Port* semnale UDF.
22H *Port* semnale stare UDF (vezi comanda de citire stare UDF).
10H Pista defectă 1, UDF0.
11H Pista defectă 2, UDF0.
18H Pista defectă 1, UDF1.
19H Pista defectă 2, UDF1.

Registrul de mod (adresă 017H)

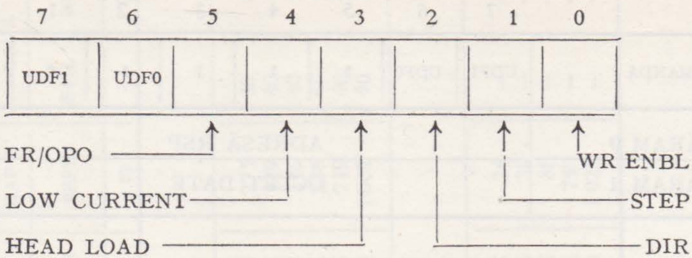
7	6	5	4	3	2	1	0
1	1		0	0	0		

Bit 1 — NR. CĂRUCIOARE; Bit 0 — MOD DMA

Bit 1 : „0" – 2 cărucioare separate
„1" – un singur cărucior pentru ambele capete
Bit 0 : „0" – mod cu DMA
„1" – mod fără DMA
La reset, biții 0,1 sînt la „0".

În mod cu DMA, se transferă date prin dialog DRQ-DACK și se lansează întreruperea la sfîrșitul transferului. În mod fără DMA, se lansează întreruperi la fiecare transfer.

Registrul port pentru semnale control UDF (adresă 023H)



La minidiscuri, bitul 5 poate fi folosit la pornire/oprire motor. Înainte de a aplica o comandă, registrul trebuie în prealabil citit, pentru a păstra restul biților nemodificați prin reînscriere.

Registrul port pentru semnale stare UDF (adresă 022H)

Configurația biților corespunde răspunsului la comanda de citire stare UDF (cod „06CH" sau „0ACH"). Citirea *port*-ului va actualiza starea NOT READY, dar nu va șterge memorarea NOT READY.

2.3.5.8. COMENZI CU PRELUCRARE DE DATE

Înaintea comenzilor de scriere/citire se produc următoarele operații pregătitoare:

— se examinează liniile de stare specifice;
— se efectuează implicit căutarea pistei;
— se încarcă discul pe capul magnetic;
— se citește BI pentru verificarea adresei pistei;
— se localizează sectorul adresat.

La sfîrșitul operației (transfer sau eroare), 8271 lansează întrerupere și umple RR cu datele corespunzătoare.

Operațiile pot fi pe unul sau pe mai multe sectoare. Dacă la un sector apare eroare CRC, operația se încheie la sectorul eronat, iar numărul sectorului respectiv se poate afla cu comanda de citire RSP la adresa 006H (număr sector curent).

În cazul operațiilor multisector, cu transferuri în mod DMA, 8257 se încarcă o singură dată cu numărul total de transferuri.

2.3.5.8.1. **Operații pe un sector**

Format:

	7	6	5 4 3 2 1	0
COMANDA	UDF1	UDF0	COD COMANDĂ	RC
PARAM 0	ADRESĂ PISTĂ			RP
PARAM 1	NUMĂR SECTOR			RP

Coduri de comandă (biții 5÷0):

12H Citire sector normal
16H Citire sector normal sau special
0AH Scriere sector
0EH Scriere sector special
1EH Verifică sector normal sau special

2.3.5.8.2. Operații multisector

Format:

	7	6	5	4	3	2	1	0
COMANDA	UDF1	UDF0	COD COMANDĂ					RC
PARAM 0	ADRESĂ PISTĂ							RP
PARAM 1	NUMĂRUL PRIMULUI SECTOR							RP
PARAM 2	LUNS			NUMĂR DE SECTOARE				RP

Coduri de comandă (pozițiile 5÷0):

13H Citire sectoare normale.
17H Citire sectoare normale sau speciale.
0BH Scriere sectoare normale.
0FH Scriere sectoare speciale.
1FH Verifică sectoare normale sau speciale.
00H Caută coincidență configurație (SCAN) în sectoare normale.
04H Caută coincidență configurație (SCAN) în sectoare normale sau speciale.

Adresa de sector, numărul de sectoare și lungimea sectorului corespund formatului de înregistrare adoptat. Biții 7÷5 din PARAM 2 corespund exponentului puterii lui 2 care, înmulțită cu 128, dă lungimea sectorului. Operațiile de citire se fac cu verificarea CRC și cu transfer în memorie. Operațiile de scriere se fac cu adăugarea octeților CRC și cu înregistrarea pe disc flexibil, în locul sectoarelor cu BI identificat. Operațiile de verificare se fac cu citirea și verificare CRC, fără transfer în memorie.

2.3.5.8.3. Operații de căutare a configurației de date (SCAN)

Format:

	7	6	5	4	3	2	1	0	
COMANDA	UDF1	UDF0	0	0	0	0/1	0	0	RC
PARAM 0	ADRESĂ PISTĂ								RP
PARAM 1	NUMĂRUL PRIMULUI SECTOR								RP
PARAM 2	LUNS			NUMĂR DE SECTOARE					RP
PARAM 3	TIP SCAN		MĂRIME PAS						RP
PARAM 4	LUNGIME CHEIE								RP

Bitul 2 al comenzii este 0/1 pentru căutarea cu sectoare normale/sectoare normale și speciale.

PARAM 0, 1, 2 stabilesc spațiul pe care se face căutarea coincidenței.

TIP SCAN (biții 7 și 6 din PARAM 3) poate fi:

00 — EQ, *Equal*, căutare cu egalitate pe toată lungimea cheii.

01 — GEQ, *Greater or Equal*, căutare pînă la găsirea unui șir $\geqslant$ decît șirul din cheie.

10 — LEQ, *Less or Equal*, căutare pînă la găsirea unui șir $\leqslant$ decît șirul din cheie.

MĂRIME PAS (biții $5 \div 0$, din PARAM 3) corespunde factorului de întrețesere, care determină ordinea în care se aleg sectoarele la căutarea coincidenței.

PARAM 4 conține lungimea șirului de octeți pe care se face comparația cu șirul cheie. Se va alege un divizor al lungimii sectorului, pentru a se preveni despărțirea șirului cheie la marginile sectoarelor.

În timpul operației de căutare a coincidenței cu o configurație dată (SCAN), cheia din memorie este comparată repetitiv, folosind DMA 8257 în *autoload*, cu datele citite de pe discul flexibil. Operația se încheie la prima îndeplinire a condițiilor specificate cu TIP SCAN sau după căutarea în toate sectoarele specificate, în caz de nereușită. La încheiere, 8271 emite întrerupere și unitate centrală execută o citire a RR pentru a afla dacă s-a găsit condiția. Ulterior, se pot examina RSP pentru a obține și alte informații despre localizarea coincidenței. 8271 nu face o căutare prin „alunecare". Dacă șirul poate îndeplini condiția, dar este decalat față de cheie, coincidența nu este semnalată. Se poate folosi, în cheie, caracterul 0FFH, care nu contează în comparare, pentru a obține divizarea corectă a lungimii sectorului prin lungimea cheii (începutul de sector să coincidă cu începutul cheii). Comparația se face octet cu octet, între caracterele din memorie (aduse prin DMA) și caracterele citite de pe discul flexibil.

2.4. CIRCUITUL LSI 8272 PENTRU ÎNREGISTRARE ÎN DENSITĂȚILE SIMPLĂ SAU DUBLĂ

Circuitul LSI 8272 este un controlor de disc flexibil perfecționat, cu multe facilități în plus față de 8271. 8272 poate opera cu pînă la 4 UDF de 8 *inch* sau 5 $^1/_4$ *inch*, în densitate simplă (mod FM) sau densitate dublă (mod MFM), în formate standard, pe ambele fețe ale dischetei. Sînt furnizate semnale care facilitează adăugarea circuitelor PLL și a circuitelor de precompensare (vezi § 3.4.). Ca și 8271, poate lucra în mod DMA (cu 8257) sau fără DMA (cu generarea unei întreruperi la fiecare transfer de octet). De notat că, în densitate dublă, se impune varianta cu DMA, deoarece intervalul mic între două transferuri de date, tipic de 16 µs (13 µs în cazul cel mai defavorabil), solicită viteză mare a microprocesorului. 8272 are posibilitatea de a executa transfer de pe aceeași pistă, de pe ambele suprafețe, la o singură comandă și poate asigura deplasarea la pista dorită, simultan, pe mai multe UDF.

2.4.1. CONEXIUNILE EXTERNE

Schema bloc a circuitului 8272, figura 2.16, este asemănătoare celei de la 8271. Capsula are 40 de conexiuni externe, figura 2.17, cu unele semnale noi, pentru realizarea avantajelor prezentate:

RST (I)★	*
$\overline{RD}$ (I)	*
$\overline{WR}$ (I)	*
$\overline{CS}$ (I)	*
A0 (I)	*
DB0 ÷ 7 (T)	*
DRQ (O)	*
DACK (I)	*
TC (I)	*Terminal Count.* Semnal de la 8257. DMA a încheiat transferul cu numărul de octeți programat.
IDX (I)	** (INDEX)
INT (O)	*
CLK (I):	Ceas 8 MHz, factor 1/2.
GND	*
WCK (O)	Impulsuri scriere cu lățime de 250 ns (FM : 500 kHz, MFM : 1 MHz).
RDW (I);	** $\overline{\text{(DATA WINDOW)}}$
RDD (I)	** $\overline{\text{(UNSEP DATA)}}$
VCO Sync (O)	* (INSYNC)

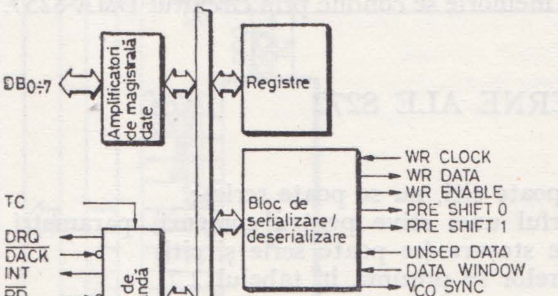


Fig. 2.16. Circuitul LSI 8272. Schema-bloc

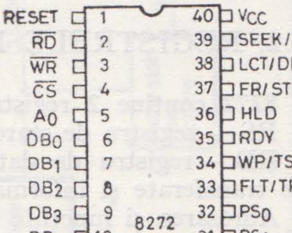


Fig. 2.17. Circuitul LSI 8272. Conexiuni externe

★ vezi nota de la pg. 139

WE (O)	* (WR ENABLE)
MFM (I)	MFM : „1", FM : „0"
SS (O)	Selecție față (fața 0 : „0", fața 1 : „1")
US0, US1 (O)	Selecție UDF0 ÷ 3, codare binară.
WDA (O)	** ($\overline{\text{WR DATA}}$)
PS1, PS2 (O)	Semnale precompensare (MFM)
FLT/TR0 (I)	Semnal multiplexat citire-scriere/căutare
	C-S : ** ($\overline{\text{FAULT}}$)
	Căutare : ** ($\overline{\text{TRK0}}$)
WP/TS (I)	C-S : ** (WR PROTECT)
	Căutare : semnal UDF cu 2 capete
RDY (I)	Multiplexarea READY0 ÷ 3 de la UDF0 ÷ 3
HLD (O)	* (HEAD LOAD)
FR/STP (O)	C-S : * (FLR)
	Căutare : * (SEEK/STEP)
LCT/DIR (O)	C-S : * (LOW CURRENT)
	Căutare : * (DIRECTION)
SEEK/RW (O)	Selectare citire-scriere/căutare
	C-S : „0"
	Căutare : „1"
VCC	Alimentare (+5 V)

Notă: *, identic cu 8271; ** aceeași semnificație ca la 8271, dar invers logic; C-S = citire-scriere.

În figura 2.18 se dă o schemă de principiu, care descrie atașarea circuitelor anexe pentru conectarea 8272 la magistrala unui microcalculator și la o UDF tipică. Transferurile la memorie se conduc prin circuitul DMA 8257.

2.4.2. REGISTRELE INTERNE ALE 8272

8272 conține 2 registre:

RS — registru de stare (se poate citi, nu se poate scrie);

RD — registru de date (vîrful unei stive pentru comenzi, parametri, date transferate și informații de stocare (se poate scrie și citi)

Adresarea și funcțiile registrelor se prezintă în tabelul 2.7

TABELUL 2.7. Adresarea și funcțiile registrelor interne ale 8272

A0	$\overline{RD}$	$\overline{WR}$	Funcție
0	0	1	RS (citire)
0	1	0	Ilegal
0	0	0	Ilegal
1	0	0	Ilegal
1	0	1	RD (citire)
1	1	0	RD (scriere)

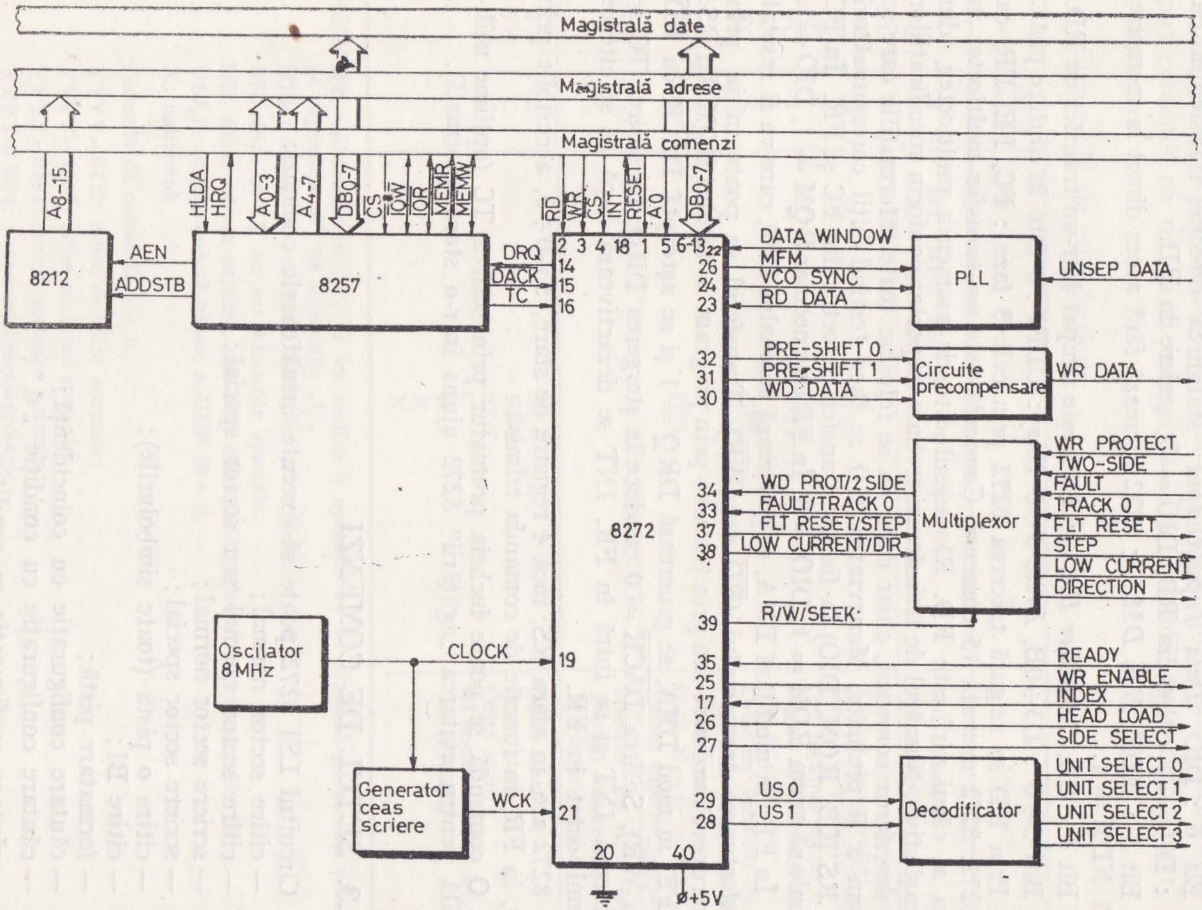


Fig. 2.18. Conectarea la magistrala unui microcalculator a unui sistem de disc flexibil cu 8272

RS conţine următoarele informaţii:

Bit 7 : RQM, *Request for Master*; RD este pregătit pentru următorul transfer; pentru transfer se analizează şi biţii 6 şi 4 (DIO şi CB).

Bit 6 : DIO, *Data Input/Output*; stabilirea direcţiei transferului cu RD : DIO = 1 citire din RD, DIO = O, scriere în RD.

Bit 5 : NDM, *Non DMA* ; 8272 lucrează fără acces direct la memorie dacă NDM = 1.

Bit 4 : CB, *Controller Busy* ; 8272 este ocupat într-un transfer cu RD.

Biţi 3÷0 : D3÷0B, *Drive 3÷0 Busy*: UDF3÷0 este în căutare pistă.

Prin RD se asigură trecerea 8272 prin cele 3 faze : FC, FE, FR, ca la 8271. 8272 execută 15 comenzi. Comanda este un transfer multioctet în stiva al cărui vîrf este RD. Răspunsul este de asemenea multioctet, din aceeaşi stivă. Numărul de transferuri trebuie respectat conform specificaţiilor din descrierea comenzii, chiar dacă nu se folosesc toate informaţiile care se impun a fi preluate. Manevrarea RD se face testînd biţii corespunzători din RS (CB, RQM, DIO) la fiecare transfer pe octet din FC şi FR. În FC trebuie să avem RQM = 1, DIO = 0. În FR este necesar RQM = 1, DIO = 1.

În FE, în mod fără DMA, se generează semnalul INT, care va fi resetat la efectuarea transferului ($\overline{RD}$ sau $\overline{WR}$). Transferul se poate iniţia prin întrerupere hardware sau prin testarea prin program a bitului RQM din RS. În FE, în mod DMA, se generează DRQ = 1 şi se aşteaptă $\overline{DACK}$ cu $\overline{RD}$ (sau $\overline{WR}$). Sosirea $\overline{DACK}$ = 0 conduce la ştergerea DRQ. La sosirea TC se activează INT şi se intră în FR. INT se dezactivează după ce se citeşte primul octet în FR.

8272 are, în afara RS, încă 4 registre de stare, ST-0÷3, accesibile prin RD, în FR, în funcţie de comanda trimisă

O comandă se poate încheia prematur prin sosirea TC (opţiune utilizabilă pentru evitarea „agăţării" 8272 ajuns într-o stare anormală).

2.4.3. SETUL DE COMENZI

Circuitul LSI 8272 poate să execute următoarele comenzi:

— citire sector normal;
— citire sector normal sau sector special;
— scriere sector normal;
— scriere sector special;
— citire o pistă (toate simbolurile);
— citire BI;
— formatare pistă;
— căutare configuraţie cu coincidenţă;
— căutare configuraţie cu condiţie „$\leqslant$" ;
— căutare configuraţie cu condiţie „$\geqslant$" ;
— recalibrare;
— citire stare întrerupere;
— specificare parametri UDF;

— citire stare UDF;
— căutare pistă.

Orice altă comandă constituie comandă invalidă și trebuie încheiată cu FR pe un octet. După comenzile de căutare pistă sau recalibrare, singura comandă ce nu va fi interpretată ca invalidă va fi citirea stării de întrerupere.

Comenzile de citire/scriere:

Faza	Citire/ scriere	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Comentarii
FC	S	MT	MFM	SK	0	Y	Y	Y	Y	Cod comandă
FC	S	X	X	X	X	X	HD	US1	US0	Cod comandă
FC	S					C				Număr de pistă (cilindru)
FC	S					H				Adresă de suprafață (cap)
FC	S					R				Număr al primului sector
FC	S					N				Cod lungime sector
FC	S					EOT				Număr al ultimului sector de pe pistă
FC	S					GPL				Lungime interval sector -BI fără octeții de sincronizare (6)
FC	S					DTL				Cînd N = 0, număr de octeți care se transferă
FE										Transfer de date între mediul magnetic și memoria principală
FR	C					ST-0				Octet stare
FR	C					ST-1				Octet stare
FR	C					ST-2				Octet stare
						C H R N				BI după execuția comenzii (CHRN)

Note: MT, *multitrack*: cu „1", se va realiza o operație ce se continuă pe aceeași pistă, de pe suprafața „0" pe suprafața „1".

MFM: cu „1" se selectează MFM, cu „0" se selectează FM.

SK, *skip*: cu „1" se sar blocurile speciale.

HD, *head*: adresa capului, așa cum corespunde adresei suprafeței, din BI

US0,1: adresa, codată binar, a UDF ($0 \div 3$).

X: indiferent

Codurile de comandă vor fi:

YYYY=0110: citire sectoare normale;

YYYY=1100: citire sectoare speciale;

YYYY=0101: scriere sectoare normale;

YYYY=1001: scriere sectoare speciale;

YYYY=0010: citire pistă întreagă, toate simbolurile de la INDEX la EOT (în acest caz MT=0).

S: octet înscris.

C: octet citit.

Comanda de citire BI:

Faza	Citire/ scriere	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Comentarii
FC	S	0	MFM	0	0	1	0	1	0	Cod comandă
FC	S	X	X	X	X	X	HD	US1	US0	Cod comandă
FE										Primul BI pe pistă este memorat în RD, poziţiile CHRN.
FR	C			ST-0						Informaţii de stare
FR	C			ST-1						Informaţii de stare
FR	C			ST-2						Informaţii de stare
FR	C			C						Conţinutul BI întîlnit în FE
FR	C			H						
FR	C			R						
FR	C			N						

Formatarea unei piste:

Faza	Citire/ scriere	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Comentarii
FC	S	0	MF	0	0	1	1	0	1	Cod comandă
FC	S	X	X	X	X	X	HD	US1	US0	Cod comandă
FC	S			N						Octeţi/sector
FC	S			SC						Sectoare/pistă
FC	S			GPL						Lungime GAP 3
FC	S			D						Octet de „umplutură" din sector
FE										Se formatează pistă
FR	3C			ST-0/1/2						Starea după FE
FR	4C			CHRN						Informaţii BI nesemnificative

Comenzile de căutare coincidenţă, SCAN:

Faza	Citire/ scriere	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Comentarii
FC	S	MT	MFM	SK	Y	Y	Y	0	1	Cod comandă
FC	S	X	X	X	X	X	HD	US1	US0	Cod comandă
FC	4S			CHRN						Primul BI
FC	S			EOT						Număr al ultimului sector de pe pistă.
FC	S			GPL						Lungime a intervalului sector-BI exclusiv octeţii de sincronizare
S	FC			STP						Ordinea în care se examinează sectoarele succesive

Faza	Citire/ scriere	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Comentarii
FE										Comparație
FR	3C	ST-0/1/2								Stare după FE
FR	4C	CHRN								BI după FE

Note: STP = 1: sectoarele se examinează unul după celălalt
STP = 2: sectoarele se examinează în mod alternat
YYY = 100: SCAN cu egalitate
YYY = 110: SCAN cu „≤"
YYY = 111: SCAN cu „≥"

Comanda de recalibrare:

Faza	Citire/ scriere	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Comentarii
FC	S	0	0	0	0	0	1	1	1	Cod comandă
FC	S	X	X	X	X	X	0	US1	US0	Cod comandă
FE										Execută recalibrarea

Comanda de sesizare a stării de întrerupere:

Faza	Citire/ scriere	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Comentarii
FC	S	0	0	0	0	1	0	0	0	Cod comandă
FR	C	ST-0								Starea după ultima
FC	C	Pista curentă (PCN)								operație de căutare

Comanda de specificare parametri UDF:

Faza	Citire/ scriere	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Comentarii
FC	S	0	0	0	0	0	0	1	1	Cod comandă
FC	S	SRT				HUT				Parametri
FC	S	HLT							ND	UDF

Note: SRT, *Step Rate Time*: interval între impulsuri STEP, programabil între 1÷16 ms cu pas de 1 ms; același pentru fiecare dintre cele 4 UDF.
HUT, *Head Unload Time*: interval cerut după execuția unor operații de citire sau scriere, programabil între 0÷240 ms, cu pas de 16 ms.
HLT, *Head Load Time*: interval în care UDF execută comanda HLD, programabil între 2 și 256 ms cu pas de 2 ms.
ND, *Non DMA Mode*: cu „1" se semnalează transfer fără DMA.

Comanda de sesizare stare UDF:

Faza	Citire/ scriere	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Comentarii
FC	S	0	0	0	0	0	1	0	0	Cod comandă
FC	S	X	X	X	X	X	HD	US1	US0	Cod comandă
FR	C	ST-3								Informații stare UDF

Comanda de căutare a unei piste:

Faza	Citire/ scriere	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Comentarii
FC	S	0	0	0	0	1	1	1	1	Cod comandă
FC	S	X	X	X	X	X	HD	US1	US0	Cod comandă
FC	S	(NCN)								Număr pistă căutată
FE										UDF execută căutarea

Comanda invalidă:

Faza	Citire/ scriere	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Comentarii
FC	S	Coduri invalide								Cod comandă
FR	C	ST-0								ST-0=080H

Pentru toate comenzile A0 = 1, iar X înseamnă indiferent poziționat, uzual „0". Comanda invalidă cere obligatoriu FR.

În timpul FC și FR, accesul la RD se face prin RS (DIO, RQM).

În FE, în mod fără DMA se emite INT la fiecare octet. Dacă microprocesorul nu servește rapid întreruperile, el poate testa RQM în RS, cu același efect. În mod DMA, nu se mai emit INT la fiecare octet din FE, 8272 generînd DRQ la fiecare transfer. 8257 răspunde cu $\overline{DACK}$, care va determina oprirea DRQ. Dacă transferurile se încheie, 8257 emite TC și 8272 va genera INT pentru intrarea în FR. La citirea RD din FR, generarea INT încetează.

De subliniat că trebuie respectat numărul și succesiunea de scrieri și citiri din RD, pentru ca stiva să fie corect adresată la următoarea comandă. Execuția unei comenzi poate fi trunchiată (terminată prematur) prin recepția TC. Aceasta este o metodă convenabilă de eliminare a „agățării" 8272 într-o stare anormală.

2.4.4. DESCRIEREA COMENZILOR

2.4.4.1. CITIREA SECTOARELOR NORMALE

Comanda începe să fie executată după încheierea recepției celor 9 octeți în FC. 8272 încarcă discul pe cap (dacă este descărcat), așteaptă intervalul HST și începe căutarea BI, pînă la coincidența numărului de sector al BI curent cu R, din grupul CHRN, după care transferă datele din sector pe magistrala microcalculatorului, octet cu octet. Încheierea transferului unui sector întreg este urmată de incrementarea R din stivă și căutarea BI corespunzător, după care se reia transferul pe magistrală cu datele din noul sector. Comanda se încheie la sosirea TC, după care 8272 oprește înlănțuirea transferului, dar continuă citirea sectorului curent, testează corectitudinea CRC și trece în FR.

Cantitatea de date ce pot fi manipulate într-o singură comandă depinde de constantele MT, MFM, N, conform tabelului 2.8.

TABELUL 2.8. **Cantitatea de informații transferate la un acces.**

MT	MFM	N	Număr maxim de transferuri (octeți)	Număr sector final
0	0	00	128 · 26 = 3 328	26/fața 0
0	1	01	256 · 26 = 6 656	26/fața 1
1	0	00	128 · 52 = 6 656	26/fața 1
1	1	01	256 · 52 = 13 312	26/fața 1
0	0	01	256 · 15 = 3 840	15/fața 0
0	1	02	512 · 15 = 7 680	15/fața 1
1	0	01	256 · 30 = 7 680	15/fața 1
1	1	02	512 · 30 = 15 360	15/fața 1
0	0	02	512 · 8 = 4 096	8/fața 0
0	1	03	1 024 · 8 = 8 192	8/fața 1
1	0	02	512 · 16 = 8 192	8/fața 1
1	1	03	1 024 · 16 = 16 384	8/fața 1

Cu opțiunea MT se pot citi ambele fețe ale dischetei (pentru UDF care permit acest lucru; CDC 9404 și MOM 6400 nu permit).

Dacă N = 0, DTL definește lungimea pe care 8272 o consideră un sector, la transfer pe magistrală. Dacă DTL este mai mic decît lungimea standard a unui sector, octeții rămași nu se transmit pe magistrală, deși 8272 citește tot sectorul fizic și verifică CRC. În funcție de TC, poate efectua și în acest caz operații multisector. Dacă N ≠ 0, DTL este un parametru nesemnificativ (dar care trebuie trimis în FC).

La încheierea FE, capul este descărcat după intervalul HUT. Dacă între timp apare o altă comandă, pe pista curentă, se elimină HST la noua comandă, iar referința de măsurare a intervalului HUT se deplasează după ultima FE.

Dacă 8272 citeşte toate datele dintr-un sector final (26/15/8) şi nu apare TC (sau microprocesorul nu a încheiat transferurile), 8272 trimite EN = 1 în ST-1 şi încheie FE. Dacă 8272 detectează INDEX de două ori, fără găsirea sectorului cerut în R, 8272 trimite ND = 1 în ST-1 şi încheie FE. După fiecare BI şi sector se testează CRC. La eroare în BI se trimite DE = 1 în ST-1, iar la eroare în sector se trimite DD = 1 în ST-2 şi se încheie FE. Dacă se citeşte un sector special şi s-a trimis SK = 0 în octetul 1 al FC, se trimite CM = 1 în ST-2 şi se termină FE după citirea sectorului. Dacă SK = 1, sectorul special se sare şi se citeşte următorul sector. În timpul transferurilor, dacă magistrala nu reia octetul disponibil în 27 μs (FM) sau 13 μs (MFM), se trimite OR = 1 în ST-1 şi se încheie FE. Dacă operaţia de citire/scriere se încheie, informaţiile despre BI din FR (CHRN) depind de MT şi EOT (vezi tabelul 2.9)

TABELUL 2.9. **CHRN după încheierea unor operaţii de citire/scriere**

MT	EOT	Ultimul sector transferat	C	H	R	N
0	01AH 00FH 008H	1÷25/faţa 0 1÷14/faţa 0 1÷7/faţa 0	NC	NC	R+1	NC
	01AH 00FH 008H	26/faţa 0 15/faţa 0 8/faţa 0	C+1	NC	01	NC
	01AH 00FH 008H	1÷25/faţa 1 1÷14/faţa 1 1÷7/faţa 1	NC	NC	R+1	NC
	01AH 00FH 008H	26/faţa 1 15/faţa 1 8/faţa 1	C+1	NC	01	NC
1	01AH 00FH 008H	1÷25/faţa 0 1÷14/faţa 0 1÷8/faţa 0	NC	NC	R+1	NC
	01AH 00FH 008H	26/faţa 0 15/faţa 0 8/faţa 0	NC	LSB	01	NC
	01AH 00FH 008H	1÷25/faţa 1 1÷14/faţa 1 1÷7/faţa 1	NC	NC	R+1	NC
	01AH 00FH 008H	26/faţa 1 15/faţa 1 8/faţa 1	C+1	$\overline{LSB}$	01	NC

Note: NC – fără schimbare

$\overline{LSB}$ – ultimul bit al H se complementează

2.4.4.2. SCRIEREA SECTOARELOR NORMALE ŞI SPECIALE

Se intră în FE după primirea celor 9 octeţi din FC. Se procedează ca şi la citire, pînă la găsirea BI corespunzător, după care se preiau octeţii de date de pe magistrală şi se trimit la UDF în format cerut de standard. După încheierea unui sector, R se incrementează şi se scrie sectorul următor, pînă la primirea TC. La sosirea TC, 8272 încheie corect sectorul curent. Dacă TC apare înainte de încheierea sectorului, octeţii rămaşi se scriu cu 00H. Dacă s-a detectat eroare CRC în căutare BI, se trimite DE = 1 în ST-1 şi se încheie FE (în ST-0 se trimit bit 7 = 0, bit 6 = 1).

Spre deosebire de citire, intervalele de timp între transferuri pot fi pînă la 31 μs în FM şi 15 μs cu MFM. Depăşirea conduce la trimiterea OR = 1 în ST-1 şi încheierea FE, cu poziţionarea în ST-0 a biţilor 7,6 = 0,1.

La comanda de scriere a sectoarelor speciale, execuţia este asemănătoare, numai că sectoarele vor fi întregistrate cu mărci speciale.

2.4.4.3. CITIREA SECTOARELOR SPECIALE

Se procedează la fel ca la scrierea sectoarelor normale, numai că la detecţia unei mărci de sector special, dacă SK = 0, se citesc şi se trimit datele, cu DM = 1 în ST-2 şi încheierea FE. Dacă SK = 1, sectorul se sare şi se citeşte următorul sector.

2.4.4.4. CITIREA COMPLETĂ A UNEI PISTE

Se procedează la fel ca la citire, numai că se transferă toate simbolurile înscrise. După detecţia semnalului INDEX, se citesc toţi octeţii din intervale, octeţii de sincronizare, mărcile, conţinutul BI şi al sectoarelor, inclusiv octeţii CRC, într-un flux continuu, chiar dacă se întîlnesc erori CRC. Se compară conţinutul primului BI cu R şi se trimite ND = 1 în ST-1, dacă apare neconcordanţă. FE se încheie după ce s-au citit EOT sectoare (EOT_{max}= = 0FFH = 255). Dacă 8272 nu găseşte nici o marcă de BI între două semnale INDEX se trimite MA = 1 în ST-1 şi se termină FE.

2.4.4.5. CITIREA UNUI BLOC DE IDENTIFICARE

Comanda furnizează poziţia curentă a capului de citire. 8272 citeşte cel mai apropiat BI. Dacă între două INDEX nu se găseşte nici o marcă de BI, se trimite MA = 1 în ST-1, iar dacă BI citit are eroare CRC, se trimite ND = 1 în ST-1. Cazurile de erori se încheie cu trimiterea în ST-0 a biţilor 7,6 = 0,1.

2.4.4.6. FORMATAREA

Se formatează conform standardelor FM (IBM System 3740) sau MFM (IBM System/34). Cîmpurile de date din BI se furnizează de către magistrală (4 cereri/sector : CHRN). Se permite astfel formatarea aleatoare (eventual cu un factor de întreţesere conform standardului). După formatarea unui

sector, registrul intern R se incrementează automat. FE se încheie la noua apariţie a semnalului INDEX de la UDF.

Dacă se recepţionează de la UDF semnalul WRITE FAULT (FLT/TR0 la 8272), se trimite EC = 1 în ST-0 şi se încheie FE. De asemenea, trecerea UDF în READY = 0, la începutul unei FE, trimite în ST-0 biţii 7,6 = 0,1.

La formatare se recomandă utilizarea parametrilor din tabelul 2.10.

TABELUL 2.10. **Parametrii formatelor standard**

Mod	Lungime sector	N	SC	GPL scriere/citire	GPL Formatare	Observaţii
FM	128	00	01AH	007H	01BH	Format IBM *Diskette* 1
	256	01	00FH	00EH	02AH	Format IBM *Diskette* 2
	512	02	008H	01BH	03AH	
	1 024	03	004H	—	—	
	2 048	04	002H	—	—	
	4 096	05	001H	—	—	
MFM	256	01	01AH	00EH	036H	Format IBM *Diskette* 2D
	512	02	00FH	01BH	054H	
	1 024	03	008H	035H	074H	Format IBM *Diskette* 2D
	2 048	04	004H	—	—	
	4 096	05	002H	—	—	
	8 192	06	001H	—	—	

2.4.4.7. COMENZI DE CĂUTARE A COINCIDENŢEI CU O CONFIGURAŢIE DATĂ (SCAN)

Se execută compararea între datele de pe dischetă şi cele furnizate pe magistrală, în mod DMA sau fără DMA. Comparaţia se face octet cu octet, cu blocarea pe un sector în care condiţia se îndeplineşte. Condiţia poate fi: cu egalitate, cu inferioritate sau egalitate faţă de magistrală, ori cu superioritate sau egalitate faţă de magistrală. Dacă la sectorul curent condiţia nu se îndeplineşte, se incrementează R cu STP şi se reia căutarea pe noul sector calculat. FE se încheie la coincidenţă, la apariţia ultimului sector de pe pistă (EOT) sau dacă s-a primit TC.

La coincidenţă se trimite SH = 1 în ST-2. Dacă nu se obţine coincidenţa între sectorul specificat în R şi EOT, se trimite SN = 1 în ST-2. Dacă se primeşte TC de la 8257, se încheie comparaţia cu octetul curent. Apariţia SH şi SN se rezumă în tabelul 2.11

Dacă apare marca de sector special (şi SK = 0), acesta este privit ca ultimul sector de pe pistă, se trimit CM = 1 în ST-2 şi se termină FE. Cu SK = 1, sectorul special este sărit şi se citeşte următorul sector (concomitent cu poziţionarea CM = 1 în ST-2). Cînd STP (01 — sectoare succesive, 02 — sectoare alternate) sau MT sînt programate, trebuie menţionat că ultimul sector de pe pistă trebuie să fie citit. Fie, de exemplu, comanda SCAN cu STP = 02, MT = 0 cu sectoarele înscrise secvenţial 1÷26 şi cu sector de start R = 21. După citirea sectoarelor 21, 23, 25, sectorul 26 se va sări şi va apărea INDEX înaintea găsirii sectorului 26, ceea ce conduce la sfîrşit anormal. Dacă s-a stabilit EOT la 25 sau dacă se startează căutarea la sectorul 20, comanda se desfăşoară normal.

TABELUL 2.11. **Semnalările SN, SH în urma operațiilor SCAN**

Comandă	ST-2		Observații
	Bit 2 = SN	Bit 3 = SH	
Coincidență cu egalitate	0 1	1 0	$D_{UDF} = D_{MAG}$ $D_{UDF} \neq D_{MAG}$
Coincidență cu inferioritate sau egalitate	0 0 1	1 0 0	$D_{UDF} = D_{MAG}$ $D_{UDF} < D_{MAG}$ $D_{UDF} > D_{MAG}$
Coincidență cu superioritate sau egalitate	0 0 1	1 0 0	$D_{UDF} = D_{MAG}$ $D_{UDF} > D_{MAG}$ $D_{UDF} < D_{MAG}$

Datele de pe magistrală pot fi furnizate de DMA sau microprocesor. Pentru a evita poziționarea în ST-1 a DR = 1, este necesar ca intervalele între transferurile pe octet să fie sub 27 μs (FM) sau 13 μs (MFM).

2.4.4.8. CĂUTAREA

Carul cu capul magnetic este deplasat la pista specificată. 8272 conține un registru de pistă curentă PCN, cu conținut accesibil în comanda de sesizare a stării de întrerupere pe care îl compară cu NCN obținut în timpul FC. Se comandă semnalele DIRECTION și STEP pentru UDF, cu respectarea SRT din comanda de specificare parametri UDF. După fiecare impuls STEP, se reia operația de comparație și așa mai departe, pînă la coincidență, cînd se trimite SE = 1 în ST-0 și se încheie FE.

În timpul FC, 8272 este în starea „ocupat", CB = 1, dar în FE, 8272 trece în starea „neocupat", CB = 0, astfel că poate primi o nouă comandă de căutare pistă la altă UDF, permițîndu-se astfel operații de căutare suprapuse în timp pe cele 4 UDF. Starea „ocupat", corespunzătoare fiecărei UDF în căutarea unei piste, se poate afla prin citirea RS. Dacă UDF este NOT READY, la începutul sau în timpul FE, se trimite NR = 1 în ST-0 și FE se încheie cu poziționarea în ST-0 a biților 7,6 = 0,1.

2.4.4.9. RECALIBRAREA UDF

Se efectuează retragerea căruciorului cu capul magnetic în poziția corespunzătoare pistei 0. Numărătorul PCN este inițializat și este controlată apariția semnalului TRACK 00 la UDF (FLT/TK0 la 8272). Se emit impulsuri STEP pînă la apariția TRACK 00, după care se poziționează SE = 1 în ST-0 și se încheie FE. Dacă, după 77 impulsuri STEP, semnalul pistei 0 nu apare, se poziționează în plus EC = 1 în ST-0 și biții 7,6 = 0,1 în ST-0, cu încheierea FE.

Observațiile cu privire la suprapunerea în timp și la semnalul READY, de la comanda de căutare, se aplică și la prezenta comandă.

2.4.4.10. SESIZAREA STĂRII DE ÎNTRERUPERE

8272 emite întrerupere la intrarea în FR din comenzile de scriere, citire, formatare și SCAN (cu excepția citirii unei piste întregi), la schimbarea stării READY la una dintre cele 4 UDF, la încheierea FE a comenzilor de recalibrare și căutare pistă și în timpul transferurilor fără DMA. Întreruperile cauzate de evenimente care pot apărea pe oricare dintre cele 4 UDF (căutări, recalibrări, schimbare READY), se tratează numai cu comanda de sesizare a stării de întrerupere. Lansarea acestei comenzi achită semnalul INT și identifică sursa întreruperii prin biții 5, 6, 7 din ST-0 avînd semnificațiile din tabel:

SE (5)	Cod (6)	INT (7)	Semnificații
0	1	1	Schimbare READY
1	0	0	Terminare normală căutare/recalibrare
1	1	0	Terminare anormală căutare/recalibrare

Comenzile de căutare pistă și recalibrare nu au FR. Comanda de sesizare a stării de întrerupere furnizează controlul pistei atinse prin PCN.

2.4.4.11. SPECIFICAREA PARAMETRILOR UDF

Se furnizează valori pentru trei numărătoare interne: HUT, pentru intervalul cît se păstrează capul încărcat după ultima FE ce a încărcat capul, între 0÷240 ms cu pas de 16 ms (00 = 0 ms, 01 = 16 ms, 02 = 32 ms etc.); SRT, pentru intervalul între impulsuri STEP consecutive, între 1÷16 ms cu pas de 1 ms (00 = 16 ms, 01 = 15 ms, 02 = 14 ms, 03 = 13 ms etc.); HLT, interval între încărcarea capului și momentul permiterii transferului, între 2÷256 ms cu pas de 2 ms (00 = 2 ms, 01 = 4 ms, 02 = 6 ms etc.). Aceste numărătoare folosesc semnalul CLK (8 MHz). Dacă se folosește 8272 conectat la un minidisc, CLK va fi de 4 MHz și intervalele obținute cu programarea de mai sus vor avea o durată dublă.

Alegerea modului fără DMA se face cu bitul ND = 1. Dacă ND = 0 transferurile pe magistrală sînt conduse de circuitul 8257.

2.4.4.12. SESIZAREA STĂRII UDF

Se obține ST-3 ce descrie interfața cu UDF (vezi tabelul 2.15).

2.4.4.13. COMANDA INVALIDĂ

Orice cod de comandă diferit de cele specificate mai sus este cod invalid. În FE, 8272 trimite în ST-0 biții 7,6 = 1,0. Se poate folosi comanda pentru a plasa 8272 în stare de așteptare. Dacă, după comenzile de recalibrare sau căutare pistă, nu se trimite o comandă de sesizare a stării de întrerupere, orice altă comandă este considerată invalidă.

2.4.5. OCTEȚII DE STARE

TABELUL 2.12. **Octetul de stare ST-0**

Bit	Nume	Simbol	Descriere
D7,6	Cod întrerupere	IC	D7 D6
			0 0 Terminare normală (NT)
			0 1 Terminare anormală (AT)
			1 0 Comandă invalidă (IC)
			1 1 Întrerupere la schimbare READY
D5	Sfîrșit de căutare	SE	8272 încheie FE căutare pistă
D4	Test UDF	EC	Apariția WRITE FAULT sau recalibrare fără succes (nu s-a detectat TRACK 00)
D3	NOT READY	NR	La comenzi de citire/scriere pe UDF NOT READY sau la o cerere a suprafeței 1 pe UDF cu o singură suprafață de înregistrare.
D2	Adresă cap	HD	Starea capului la întrerupere
D1,0	Cod selecție	US0,1	UDF care a lansat întrerupere

TABELUL 2.13. **Octetul de stare ST-1**

Bit	Nume	Simbol	Descriere
D7	Sfîrșit pistă	EN	8272 caută un sector după ultimul sector de înregistrat
D6			Neutilizat („0")
D5	Eroare de date	DE	Eroare CRC în BI sau sector
D4	Eroare de ritm	OR	Magistrala nu a transferat octetul în intervalul de timp impus
D3			Neutilizat („0")
D2	Lipsă date	ND	La citire/scriere/SCAN care presupun căutare sector, nu a fost găsit BI pentru sectorul specificat
			La comanda citire BI, nu a existat pe pistă nici un BI fără eroare CRC
			La comanda citire pistă întreagă nu a fost găsit sectorul de start specificat
D1	Scriere imposibilă	NW	În comenzi de scriere/formatare se detectează semnalul $\overline{\text{WRITE PROTECT}}$ (WP/TS) activ

TABELUL 2.13. (continuare)

Bit	Nume	Simbol	Descriere
D0	Lipsă marcă	MA	8272 nu găsește nici o marcă de BI (0FEH/0C7H) între două semnale INDEX
			8272 nu găsește marca de sector (0FEH/0C7H sau 0F8H/0C7H). Se asociază cu MD = 1 în ST-2.

TABELUL 2.14. Octetul de stare ST-2

Bit	Nume	Simbol	Descriere
D7			Neutilizat („0")
D6	Marcă specială	CM	În citire sectoare normale sau SCAN, 8272 a întîlnit un sector special
D5	Eroare în date	DD	Eroare CRC în cîmpul de date al sectorului
D4	Eroare pistă	WC	Numărul de pistă din BI citit nu coincide cu cel cerut. Se asociază cu ND = 1 în ST-1.
D3	Coincidență cu egalitate	SH	În SCAN s-a respectat condiția de coincidență cu egalitate
D2	Condiție de SCAN nesatisfăcută	SN	În SCAN nu s-a respectat condiția impusă
D1	Pistă defectă	BC	Numărul de pistă din BI diferă de cel cerut și este 0FFH
D0	Lipsă marcă sector	MD	La citire sector lipsește marca 0FBH/0C7H sau 0F8H/0C7H

TABELUL 2.15. Octetul de stare ST-3

Bit	Nume	Simbol	Semnal pe interfața UDF–8272
D7	Defect	FT	WRITE FAULT – FLT/TK0
D6	Protecție scriere	WP	WR PROTECT – WP/TS
D5	READY	RY	READY – RDY
D4	Pista 0	T0	TRACK 00 – FLT/TKO
D3	Două fețe	TS	(DOUBLE SIDE) – WP/TS
D2	Adresă cap	HD	(SEL HEAD) – SS
D1,0	Selecție UDF	US1,0	Cod selecție UDF0 ÷ 3

Observație: Semnalele între paranteze nu există pe interfețele CDC 9404 sau MOM 6400.

2.4.6. PROGRAMAREA CIRCUITULUI 8272

Circuitul LSI 8272 este conceput să funcționeze ca periferic al microprocesoarelor din seria 8080, dar poate fi adaptat și unei magistrale de tip Z80. Desfășurarea operațiilor constă în trecerea succesivă prin fazele de inițializare, FI, de comandă, FC, de execuție, FE și de rezultat, FR. Este necesară și legătura conversațională cu circuitele de acces direct la memorie, DMA, și de întrerupere, IT. În funcție de specificul operației se deosebesc 4 căi diferite de desfășurare a execuției, ca în figura 2.19.

Pentru comanda inițială de programare a 8272 cu caracteristicile UDF, se desfășoară numai fazele FI și FC. Acest tip de comandă se încheie fără trecerea prin fazele FE și FR, figura 2.19 *a*. Pentru execuția comenzii de sesizare a stării UDF, nu este necesară faza FE, după FC urmînd imediat FR, figura 2.19*c*. Pentru comenzile de căutare sau recalibrare, de tip *paralel*, figura 2.19*b*, se intră în FE imediat după FC, dar în FE poate fi acceptată o altă comandă de căutare pe alt UDF, chiar dacă deplasarea carului UDF apelate inițial este încă în curs. La comenzi care implică transferuri de date, se intră în FE după FC, numai după ce se primește prima cerere DMA, figura 2.19 *d*. Toate operațiile, cu excepția celei de inițializare, se încheie cu lansarea unei întreruperi, la conexiunea exterioară INT a 8272. Rutina de tratare a întreruperii trebuie să asigure trecerea prin FR.

Intrarea în FC se face diferit în funcție de natura comenzii de executat, conform figurii 2.20. Dacă este tratată o comandă de tip *paralel*, se testează dacă 8272 este ocupat cu execuția unei operații cu caracter serial sau dacă execută o operație de căutare pe aceeași UDF. Dacă nici una dintre aceste condiții nu este prezentă, se poate intra în FC pentru execuția comenzii în curs, chiar dacă, eventual, sînt în desfășurare una sau mai multe operații de căutare pe alte UDF. Operațiile cu caracter *serial* implică elibe-

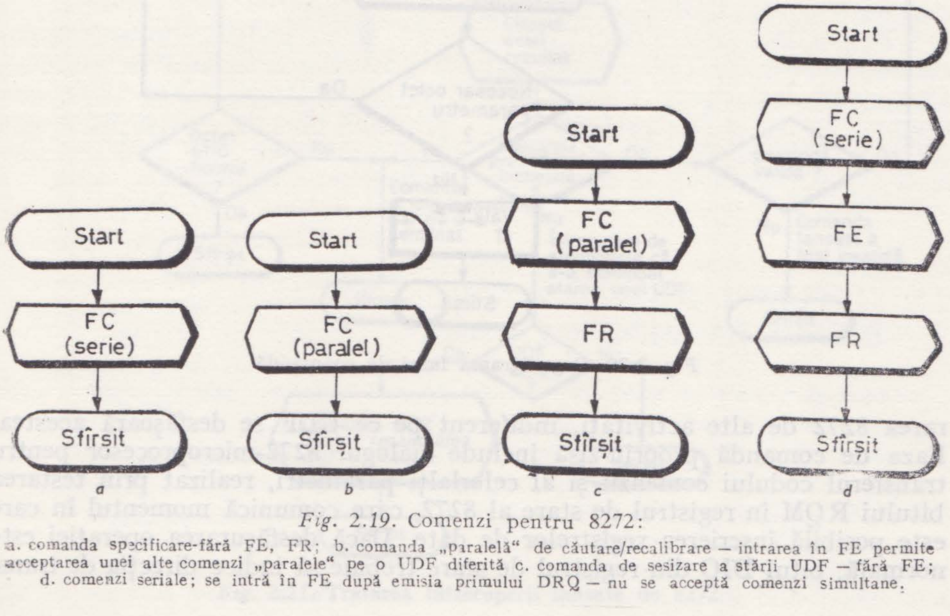


Fig. 2.19. Comenzi pentru 8272:

a. comanda specificare-fără FE, FR; b. comanda „paralelă" de căutare/recalibrare – intrarea în FE permite acceptarea unei alte comenzi „paralele" pe o UDF diferită c. comanda de sesizare a stării UDF – fără FE; d. comenzi seriale; se intră în FE după emisia primului DRQ – nu se acceptă comenzi simultane.

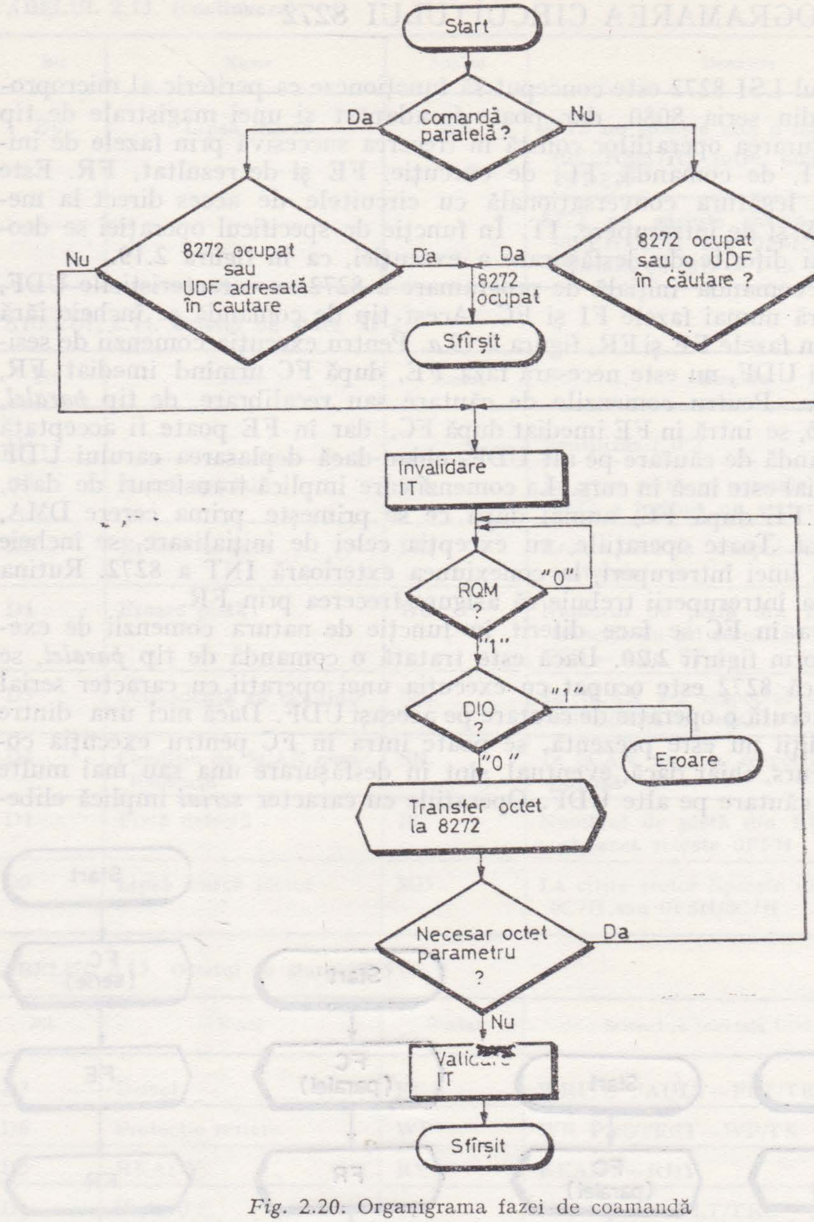


Fig. 2.20. Organigrama fazei de coamandă

rarea 8272 de alte activități, indiferent pe ce UDF se desfășoară acestea. Faza de comandă propriu-zisă include dialogul 8272-microprocesor pentru transferul codului comenzii și al celorlalți parametri, realizat prin testarea bitului RQM în registrul de stare al 8272, care comunică momentul în care este posibilă înscrierea registrelor de date. Dacă desfășurarea operației este normală, bitul DIO din registrul de stare trebuie să indice direcția de trans-

fer dinspre procesor spre controlor. Cînd DIO indică direcția contrară, se semnalează eroare. Dacă sînt realizate condițiile, transferul octetului de comandă are loc. Pentru parametrii următori dialogul prin RQM și DIO se repetă. Secvența de transfer al codului de comandă și parametrilor aferenți trebuie protejată pentru a nu fi întreruptă, prin încadrare între instrucțiunile DI și EI.

După desfășurarea, în condiții normale sau nu, a fazei de execuție, 8272 semnalează încheierea ei prin lansarea hardware a unei întreruperi. Rutina de tratare a întreruperii (figura 2.21) trebuie să preia datele referitoare la modul de desfășurare a operației aferente și să continue dialogul, pentru a lăsa controlorul într-o stare în care să poată prelua o altă comandă. Rutina va testa în prealabil dacă 8272 este sau nu ocupat cu o altă operație. Dacă nu este ocupat, înseamnă că întreruperea s-a lansat după o ope-

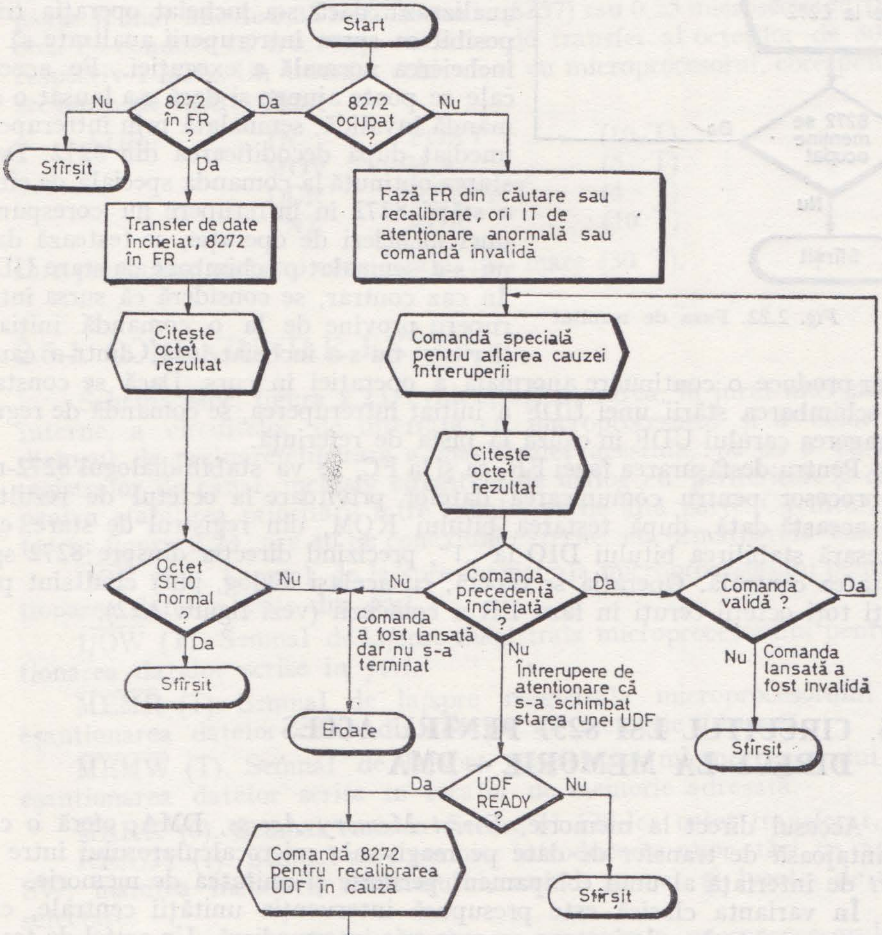


Fig. 2.21. Tratarea întreruperii lansate de 8272

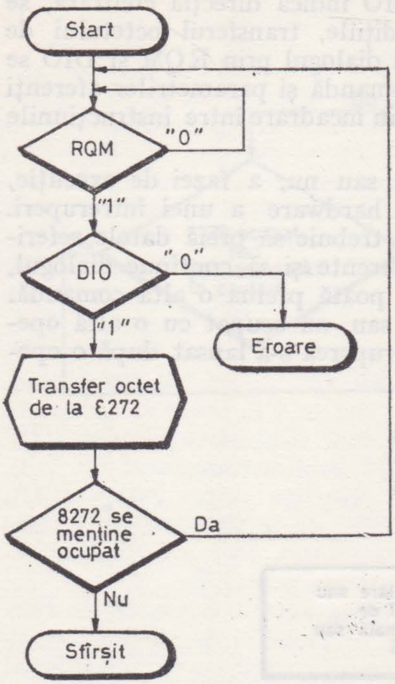


Fig. 2.22. Faza de rezultat

rație de tip *paralel* (căutare sau recalibrare) sau a apărut în mod neașteptat, ca urmare a unei schimbări a stării unei UDF sau apariției unei stări anormale. Dacă 8272 este ocupat, întreruperea provine de la o operație de tip *serial* – scriere sau citire – care s-a încheiat și a ajuns în faza FR. În cazul din urmă, se desfășoară faza FR, care stabilește dacă a apărut sau nu eroare. Cînd întreruperea a apărut din operații de căutare-recalibrare sau schimbare de stare UDF, este necesară lansarea unei comenzi speciale de citire a stării 8272 în întrerupere. După trecerea prin faza FR a acestei comenzi, se analizează dacă s-a încheiat operația, fiind posibil ca sursa întreruperii analizate să fie încheierea normală a execuției. Pe aceeași cale se poate ajunge și dacă s-a lansat o comandă invalidă, semnalată prin întrerupere, imediat după decodificarea din 8272. Dacă starea obținută la comanda specială de citire a stării 8272 în întrerupere nu corespunde unei încheieri de operație, se testează dacă nu s-a semnalat o schimbare de stare UDF. În caz contrar, se consideră că sursa întreruperii provine de la o comandă inițială, dar care nu s-a încheiat încă, dintr-o cauză ce ar produce o continuare anormală a operației în curs. Dacă se constată că schimbarea stării unei UDF a inițiat întreruperea, se comandă de regulă retragerea carului UDF în cauză la pista de referință.

Pentru desfășurarea fazei FR, ca și la FC, se va stabili dialogul 8272-microprocesor pentru comunicarea datelor privitoare la octetul de rezultat. De această dată, după testarea bitului RQM, din registrul de stare, este necesară stabilirea bitului DIO la „1", precizînd direcția dinspre 8272 spre unitatea centrală. Operația se repetă, cu același dialog, pînă cînd sînt preluați toți octeții ceruți în faza FR a comenzii (vezi figura 2.22).

2.5. CIRCUITUL LSI 8257 PENTRU ACCES DIRECT LA MEMORIE – DMA

Accesul direct la memorie, *Direct Memory Access*, DMA, oferă o cale avantajoasă de transfer de date pe magistrala microcalculatorului între un *port* de interfață al unui echipament periferic și unitatea de memorie.

În varianta clasică este presupusă intervenția unității centrale, care folosește registrele sale interne ca memorie intermediară. Un astfel de transfer periferic/memorie presupune două faze: transfer *port*/memorie în acumulatorul microprocesorului și apoi încă un transfer din registrul intern al microprocesului în memorie/*port*. Accesul direct la memorie, realizat de un

circuit LSI specializat, elimină microprocesorul de pe magistrala microsistemului, dirijînd în mod independent transferurile periferic — memorie prin intermediul semnalelor de citire/scriere, semnale generate simultan atît pentru *port*, cît și pentru memorie. În plus, circuitul DMA furnizează adresele de memorie la locații succesive pentru transferuri pe șir de caractere și semnalele de selecție pentru *port*. Accesul la magistrală se realizează în urma unui dialog cu microprocesorul folosind semnalele HOLD/HLDA pentru 8080 sau $\overline{\text{BUSRQ/BUSAK}}$ pentru Z80. Transferurile succesive se pot realiza în pachet, la viteza maximă, sau cu cîte o cerere de transfer pentru fiecare octet. În acest din urmă caz, microprocesorul poate lucra în pauzele dintre transferuri.

Circuitul 8257 oferă funcția DMA, simultan pe patru canale. Viteza maximă corespunde transferului în pachet a cîte unui octet la fiecare 4T, unde T este perioada semnalului de sincronizare al microprocesorului. T nu poate fi mai mic de 0,32 microsecunde (8257) sau 0,25 microsecunde (8257-5), ceea ce conduce la frecvențe maxime de transfer al octeților de 800 kHz, respectiv 1 MHz. Un transfer echivalent cu microprocesorul, corespunde unei secvențe de instrucțiuni:

```
ADR:   IN     PORT      (10 T)
       MOV    M,A       (5 T)
       DCR    REGCNT    (5 T)
       JNZ    ADR       (10 T)
```

care presupune un timp considerabil mai mare (30 T).

2.5.1. CONEXIUNILE EXTERNE

Schema bloc, figura 2.23, evidențiază așezarea, în jurul unei magistrale interne, a circuitelor de interfață cu microprocesorul și a celor pentru dialogul de preluare/eliberare a magistralei acestuia, pe de o parte, și a registrelor de canal, inclusiv circuitele de dialog cu perifericele și circuitele pentru stabilirea priorității între canale, pe de altă parte. Circuitul, prezentat în figura 2.24, are 40 de conexiuni externe, cu următoarele semnificații:

$\overline{\text{I/OR}}$ (T)*. Semnal de la/spre magistrala microprocesorului pentru eșantionarea datelor citite din *port*.

$\overline{\text{I/OW}}$ (T). Semnal de la/spre magistrala microprocesorului pentru eșantionarea datelor scrise în *port*.

$\overline{\text{MEMR}}$ (T). Semnal de la/spre magistrala microprocesorului pentru eșantionarea datelor citite din locația de memorie adresată.

$\overline{\text{MEMW}}$ (T). Semnal de la/spre magistrala microprocesorului pentru eșantionarea datelor scrise în locația de memorie adresată.

MARK (0). Semnal emis la fiecare al 128-lea octet transferat.

READY (I). Semnal care permite introducerea unor stări de așteptare, SW, înaintea încheierii transferului, avînd *port*-ul și locația de memorie selectate.

HLDA (I). Semnal primit de pe magistrala microprocesorului, semnificînd că aceasta s-a eliberat, în vederea transferurilor DMA.

---

* vezi nota de la pg. 139

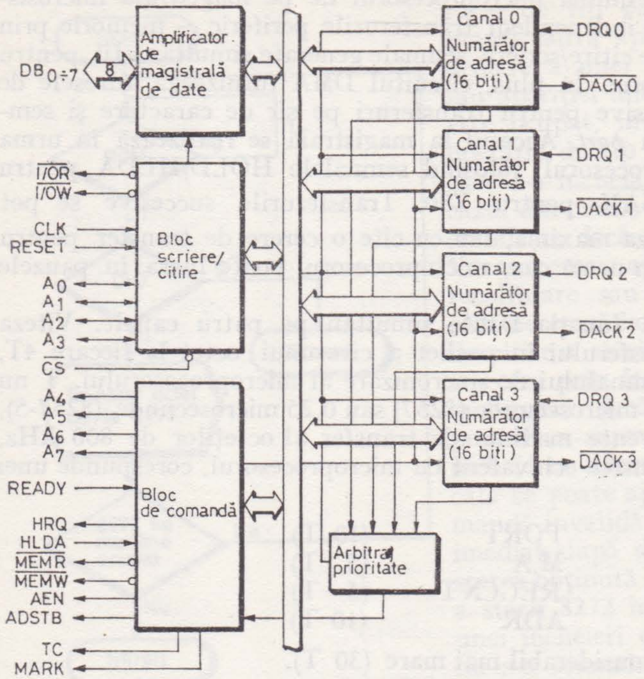


Fig. 2.23. Circuitul LSI 8257. Schema bloc

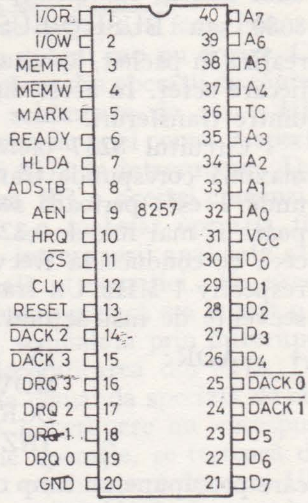


Fig. 2.24. Circuitul LSI 8257. Conexiuni externe

ADDSTB (O). Semnal de eșantionare pentru încărcarea unui *port* adițional tip 8212 cu octetul cel mai semnificativ al adresei de memorie.

AEN (O). Semnal care semnifică ocuparea magistralei microprocesorului pentru accese DMA. Validează emiterea octetului cel mai semnificativ de adresă din portul adițional.

HRQ (O). Semnal de cerere a magistralei, în vederea eliminării microprocesorului.

$\overline{\text{CS}}$ (I). Semnal de selecție pentru *port*-urile circuitului 8257. Invalid în timpul ciclilor DMA.

CLK (I). Semnal de sincronizare. Coincide cu semnalul Φ2TTL (8080) sau Φ (Z80).

RESET (I). Semnal de inițializare. 8257 intră în starea SI.

$\overline{\text{DACK 2}}$ (O). Semnal de selecție pentru portul corespunzător canalului 2. Se emite după obținerea magistralei în vederea selectării *port*-ului în vederea efectuării transferului.

$\overline{\text{DACK 3}}$ (O). Idem, pe canalul 3.

DRQ 3 (I). Semnal prin care se semnalizează circuitului DMA că este necesar un transfer pe canalul 3.

DRQ 2 (I). Idem, pe canalul 2.

DRQ 1 (I). Idem, pe canalul 1.

DRQ 0 (I). Idem, pe canalul 0.

GND. Masa tensiunii de alimentare (0V).

$(D_7 \div D^0)$ (T). Semnale de date pe magistrala microcalculatorului.

$\overline{\text{DACK 1}}$ (O). Semnal identic cu $\overline{\text{DACK 2}}$, pentru canalul 1.

$\overline{\text{DACK 0}}$ (O). Idem, canalul 0.

VCC Sursa de alimentare (+5V).

A0 ÷ 7 (T). Octetul cel mai puțin semnificativ al liniilor de adresare ale magistralei microcalculatorului.

TC(O). Semnalează sfîrșitul transferului șirului de octeți. Se emite în timpul ultimului transfer.

2.5.2. REGISTRELE INTERNE ALE 8257

8257 conține cîte 2 registre bidirecționale de 16 biți pentru fiecare dintre cele 4 canale și încă două registre unidirecționale, pentru comanda și examinarea funcționării circuitului. Semnificația fiecărei poziții din registre rezultă din tabelul 2.16.

TABELUL 2.16. **Registrele de canal, mod și stare ale 8257***

Registru	Octet	Adresa $A_3 \div A_2$	Bistabil intern	Conținut D7	D6	D5	D4	D3	D2	D1	D0
Adresă-canal 0	Cmps	0000	0	A7	A6	A5	A4	A3	A2	A1	A0
	Cms	0000	1	A15	A14	A13	A12	A 11	A10	A9	A8
Numărător de octeți-canal 0	Cmps	0001	0	C7	C6	C5	C4	C3	C2	C1	C0
	Cms	0001	1	S	C	C13	C12	C11	C10	C9	C8
Adresă-canal 1	Cmps	0010	0	A7	A6	A5	A4	A3	A2	A1	A0
	Cms	0010	1	A15	A14	A13	A12	A11	A10	A9	A8
Numărător de octeți-canal 1	Cmps	0011	0	C7	C6	C5	C4	C3	C2	C1	C0
	Cms	0011	1	S	C	C13	C12	C11	C10	C9	C8
Adresă-canal 2	Cmps	0100	0	A7	A6	A5	A4	A3	A2	A1	A0
	Cms	0100	1	A15	A14	A13	A12	A11	A10	A9	A8
Numărător de octeți-canal 2	Cmps	0101	0	C7	C6	C5	C4	C3	C2	C1	C0
	Cms	0101	1	S	C	C13	C12	C11	C10	C9	C8
Adresă-canal 3	Cmps	0110	0	A7	A6	A5	A4	A3	A2	A1	A0
	Cms	0110	1	A15	A14	A13	A12	A11	A10	A9	A8
Numărător de octeți-canal 3	Cmps	0111	0	C7	C6	C5	C4	C3	C2	C1	C0
	Cms	0111	1	S	C	C13	C12	C11	C10	C9	C8
Mod	—	1000	—	AL	TCS	EW	EP	EN3	EN2	EN1	EN0
Stare	—	1000	—	0	0	0	UP	TC3	TC2	TC1	TC0

* Simboluri folosite:

Cmps = cel mai puțin semnificativ octet;
Cms = cel mai semnificativ octet;
C = citire *port*, generează $\overline{\text{I/OR}}$, $\overline{\text{MEMW}}$;
S = scriere *port*, generează I/OW, MEMR;
AX = bit adresă de start transfer, X = 0 ÷ 15;
CX = bit de inițializare a numărătorului de octeți, X = 0 ÷ 13.

2.5.2.1. REGISTRELE DE CANAL

Fiecăruia dintre cele 4 canale îi corespunde un registru pentru adresa de memorie RAM de la care începe transferul, într-un spațiu de 64 Kocteți și încă un registru de 16 biți, ale cărui prime 14 locații au funcția de numărător, pentru pînă la 16 Kocteți, iar celelalte 2 locații, biții 14 și 15, monitorizează generarea semnalelor $\overline{I/OR}$, $\overline{I/OW}$, $\overline{MEMR}$, $\overline{MEMW}$. Adresa și numărul de octeți evoluează la fiecare transfer DMA pe canalul respectiv, iar ultimul transfer este semnalat ca încheiere a transferului, *Terminal Count*, TC, pe o conexiune exterioară a circuitului. Semnalele de citire/scriere corespund combinațiilor:

	S	C	$\overline{MEMR}$	$\overline{MEMW}$	$\overline{I/OR}$	$\overline{I/OW}$
Vîlcare	0	0	„1"	„1"	„1"	„1"
Citire *port*	0	1	„1"	„0"	„0"	„1"
Scriere *port*	1	0	„0"	„1"	„1"	„0"
Invalid	1	1	—	—	—	—

Fiecare registru de 16 biți este accesibil printr-o dublă adresare la cîte 8 biți, folosind un bistabil intern, care selectează între octetul cel mai puțin semnificativ și cel mai semnificativ.

2.5.2.2. REGISTRUL DE MOD

Registrul asigură validarea canalelor în lucru și a 4 opțiuni, asupra modului de desfășurare a dialogului pe magistrală.

7	6	5	4	3	2	1	0
AL	TCS	EW	RP	EN3	EN2	EN1	EN0

Registrul este înscris, de regulă, după completarea celor 2 registre de 16 biți ale fiecărui canal cu care se lucrează. La RESET registrul de control este șters, inhibîndu-se toate canalele și opțiunile, astfel prevenindu-se transferuri DMA nedorite la aplicarea tensiunii.

EN0 ÷ 3: *Enable Channel*, corespunde validării canalului pe care se dorește să se efectueze transferuri. Se pot face transferuri pe 1 ÷ 4 canale. Pentru mai mult de un canal, ordinea de servire corespunde unei scheme de priorități, stabilite de bitul 4.

RP: *Rotating Priority*, corespunde stabilirii schemei de priorități, între două variante: prioritate fixă sau prioritate circulară. În varianta fixă se alocă o prioritate maximă canalului 0, prioritate care descrește pînă la a fi minimă pentru canalul 3. În varianta circulară, se alocă dinamic nivelul de prioritate al fiecărui canal, după fiecare ciclu de transfer. În acest caz, fiecare canal servit va trece ultimul pe lista de priorități, în arbitrarea ciclului DMA următor.

Alegerea modului RP previne fenomenul de „gîtuire", *bottleneck*, cînd unul dintre canale monopolizează transferurile în mod DMA. Dacă sînt prezentate cereri de transfer pe mai multe canale, se vor servi toate cana

ele, pe rînd. Primul transfer se va executa conform schemei fixe de priorități.

EW: *Extended Write*, contribuie la generarea anticipată a semnalelor de scriere, pentru compensarea unor eventuali timpi de acces mai lungi ai participanților la transfer. În absența acestei opțiuni, ar fi fost necesare stări de așteptare, folosind semnalul READY de la conexiunea exterioară 6, ceea ce ar fi micșorat viteza de lucru.

TCS: *Terminal Count Stop*, specifică oprirea sau continuarea transferurilor DMA, din momentul cînd s-a emis „1" pe conexiunea exterioară TC, simultan cu poziționarea bitului corespunzător canalului, în registrul de stare. Bitul de validare a canalului este șters, iar operațiunile DMA pe acest canal se pot relua după o reînscriere în registrul de control. În cazul în care nu se programează bitul TCS, transferurile DMA continuă, la adrese succesive, cît timp perifericul emite DRQ.

AL: *Autoload*, autoîncărcare, permite, cînd este emis, să se folosească înlănțuirea de comenzi, fără intervenția microprocesorului. Opțiunea se aplică numai canalului 2, utilizînd canalul 3 ca registru cu parametri de reinițializare. Canalul 2 se programează obișnuit, pentru primul set de parametri, iar canalul 3 se programează cu setul de parametri pentru transferul ulterior. După ce se efectuează ciclurile DMA conform programării inițiale a canalului 2, se efectuează automat o trecere a parametrilor din canalul 3 în canalul 2. Opțiunea TCS nu are efect, dacă a fost prevăzută simultan cu opțiunea AL. În cazul existenței opțiunii AL, programarea canalului 2 este automat duplicată și în registrele canalului 3, ceea ce permite operații repetate pe bloc, prin programarea unui singur canal. Se pot înscrie parametri separați pentru cele două canale, dacă se programează întîi canalul 2 și apoi canalul 3. De observat că, dacă este validat și canalul 3 și apar cereri DRQ pe acest canal, se pot efectua transferuri simultane pe canalele 2 și 3, însă, la operația de reîncărcare a canalului 2, se vor încărca parametrii modificați ai canalului 3, cu o diferență dată de numărul de transferuri DMA ce s-au efectuat pe acest din urmă canal. La intrarea într-un ciclu, după reinițializarea canalului 2, se poziționează bitul UP în registrul de stări. Reinițializarea nu distruge parametrii canalului 3.

Reinițializarea se efectuează la următorul ciclu DMA după un ciclu TC, care va fi astfel primul ciclu cu noii parametri. Bitul UP din registrul de stare se va șterge după acest prim ciclu DMA. În operații înlănțuite, bitul UP se poate examina în microprocesor pentru a semnala reinițializarea și a permite înscrierea noilor parametri în canalul 3, pentru următorul bloc ce se va transfera pe canalul 2.

2.5.2.3. REGISTRUL DE STARE

Registrul indică pe care dintre canale s-a terminat transferul numărului de octeți programat, inclusiv semnalarea stării de început a unui bloc de transfer cu parametri reinițializați (numai pentru canalul 2).

7	6	5	4	3	2	1	0
0	0	0	UP	TC3	TC2	TC1	TC0

TC0÷3: *Terminal Count,* sfîrşit de numărare, corespunde stării conexiunii exterioare TC, numai că aici se specifică şi numărul de ordine al canalului pe care a avut loc evenimentul. Biţii rămîn înscrişi pînă la prima citire a registrului de stare, sau pînă la RESET. Deoarece, între citirea stării şi înscrierea de parametri, poate apărea semnalarea TC, este contraindicată reprogramarea canalelor în funcţie de semnalările din registrul de stare, dacă între cele două operaţii succesive nu se blochează transferurile DMA.

UP: *Update,* este emis la primul transfer DMA efectuat conform unor parametri de reiniţializare ai canalului 2. UP nu se şterge la citire. Se foloseşte pentru a preveni încărcarea unor parametri în canalul 3, înainte ca parametrii precedenţi să fie trecuţi în canalul 2.

2.5.3. EFECTUAREA TRANSFERURILOR DMA CU 8257

Funcţionarea 8257 corespunde diagramei de stări date în figura 2.25. În urma semnalului RESET se intră în starea SI, în care se aşteaptă lansarea unei cereri de transfer DRQ de la periferice. Ieşirea din SI şi trecerea în S0, se face la primirea DRQ pe unul din canale şi lansarea cererii HOLD, pentru obţinerea magistralei microcalculatorului. În starea S0 se aşteaptă eliminarea microprocesorului de pe magistrală, semnalată prin sosirea semnalului HLDA, după care se intră în starea S1. În acest moment, 8257 emite adresele superioare pe liniile de date, memorate în *port*-ul adiţional 8212, cu semnalul ADDSTB. Se obţin astfel toate cele 16 linii de adresă pe magistrală şi se intră în starea S2. În S2 începe transferul prin activarea operaţiei de citire şi a unei eventuale scrieri anticipate. Se emite DACK pentru dialogul DRQ-DACK şi se intră în starea S3, în care se activează scrierea şi se fac semnalările MARK şi TC, dacă este cazul. Din această stare se poate intra într-o stare de aşteptare SW, dacă nu s-a primit READY. Testarea conexiunii externe READY este eliminată dacă pe canalul respectiv a fost programată o operaţie de verificare, VERIFY. Din S3 se intră în S4, dacă s-a primit semnalul READY în operaţiile de transfer efectiv. Din SW se trece în S4 dacă se primeşte READY. Starea S4 încheie un ciclu DMA, inactivînd DACK şi, dacă este cazul, TC şi MARK. În cazul operaţiilor cu TCS, se şterge validarea canalului respectiv, dacă a apărut TC. Se stabilesc priorităţile pentru următorul ciclu DMA. Se testează DRQ şi HLDA. Dacă DRQ nu se mai emite, se şterge HOLD după semnalarea căderii HLDA. Dacă DRQ se emite în continuare, se păstrează HOLD şi, în cazul în care şi HLDA se menţine, se trece în modul de transfer *burst,* în pachet, prin succesiunea de stări S1, S2, S3, S4, S1 etc., evitîndu-se dialogul de conectare/deconectare la magistrală. La pierderea magistralei, situaţie semnalată de căderea HLDA, sau dacă perifericul nu mai cere transfer, cu DRQ la „0", se şterge HOLD şi se trece în starea SI pentru ciclul de reconectare SI, S0, S1, S2, S3, S4 etc.

În rezumat, transferul octet cu octet se desfăşoară după cum urmează (figura 2.25): perifericul cere transfer prin emiterea semnalului DRQ pe unul dintre canale; dacă este prioritar şi validat, canalul începe operaţiunea de transfer, lansînd HOLD; la recepţia HLDA se lansează DACK pe canalul

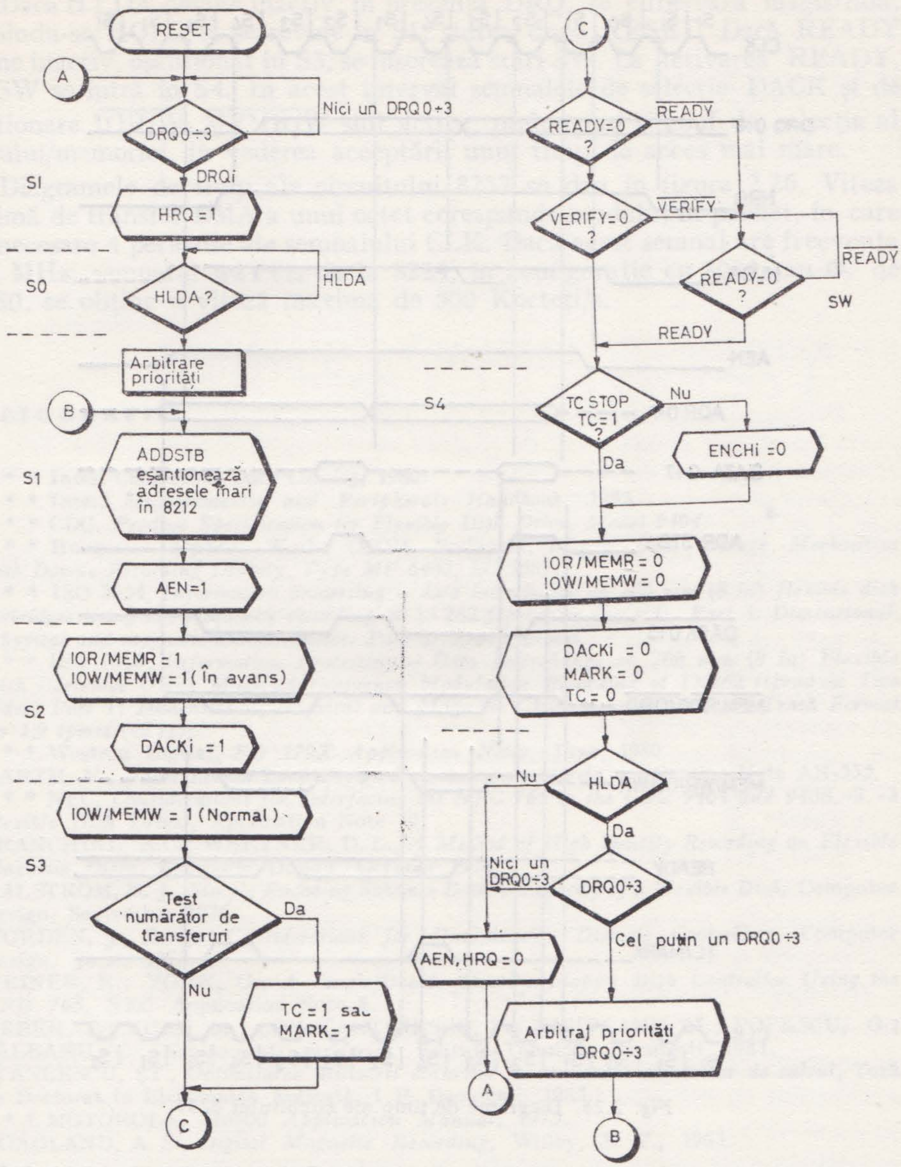


Fig. 2.25. Organigrama de funcţionare a 8257

corespunzător, pentru selectarea registrului de date al perifericului; se generează apoi perechea de semnale de eşantionare $\overline{IOR}$, $\overline{MEMW}$ sau $\overline{IOW}$, $\overline{MEMR}$, în funcţie de sensul transferului, dacă nu este specificată operaţia VERIFY; memoria este adresată la locaţia specificată în registrul de adrese al canalului, direct de 8257 pe octetul cel mai puţin semnificativ şi prin intermediul *port*-ului adiţional 8212, pe octetul cel mai semnificativ. După transferul octetului,

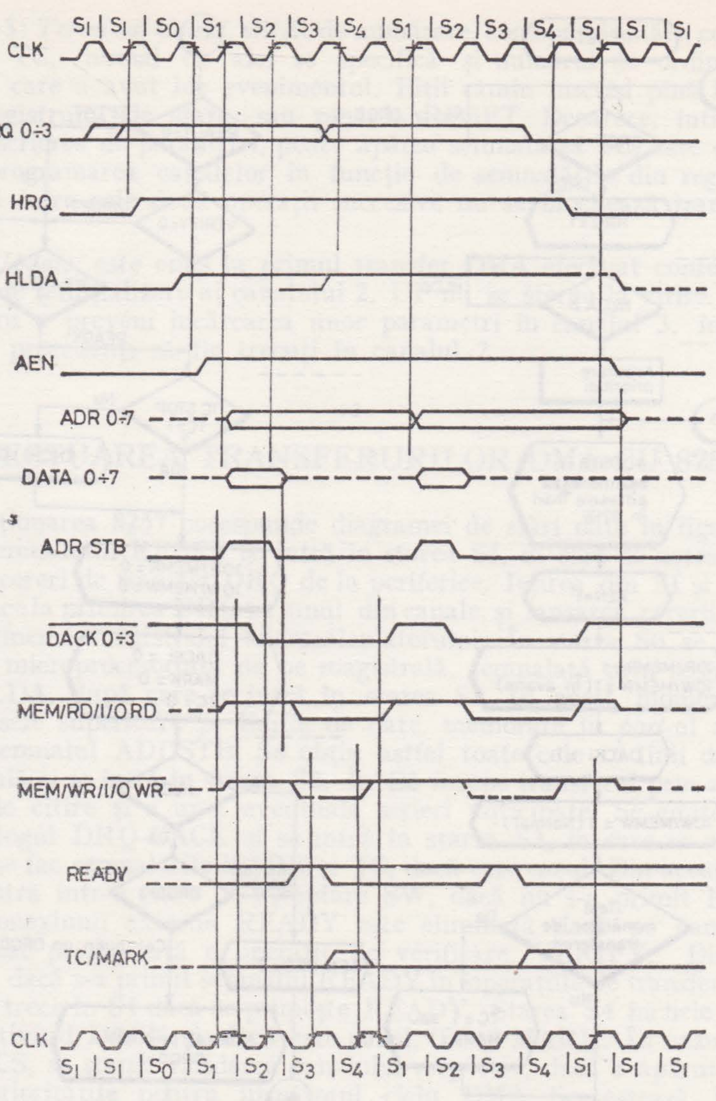


Fig. 2.26. Diagrame de timp ale circuitului 8257

semnalele de eșantionare și achitarea cererii, DACK, se inhibă, iar dacă DRQ nu se mai emite înainte de S4, se eliberează magistrala și se inhibă HOLD.

Funcționarea în mod *burst* se alege păstrînd DRQ și HLDA după efectuarea transferului, caz în care se incrementează adresa memoriei și se reiau operațiile pe octet, fără deconectare/reconectare intermediară.

Dacă se cer transferuri simultan pe mai multe canale, procedura corespunde transferului în pachet, cu comutarea canalului pe care se emit semnalele DRQ, DACK, în funcție de schema de prioritate programată. Canalul prioritar este stabilit în S4, în funcție de semnalele DRQ prezente.

Dacă HLDA devine inactiv, în prezența DRQ, se eliberează magistrala, inhibîndu-se HOLD și se revine în SI, starea după RESET. Dacă READY devine inactiv, eșantionat în S3, se inserează stări SW. La activarea READY, din SW se intră în S4. În acest interval semnalele de selecție DACK și de eșantionare $\overline{IOR/W}$, $\overline{MEMR/W}$ sînt active, prelungind timpul de selecție al *port*-ului/memoriei, în vederea acceptării unui timp de acces mai mare.

Diagramele de timp ale circuitului 8257 se dau în figura 2.26. Viteza maximă de transfer DMA a unui octet corespunde modului în pachet, în care sînt necesare 4 perioade ale semnalului CLK. Dacă acest semnal are frecvența de 2 MHz, semnalul Φ2TTL de la 8224, în configurație cu 8080 sau Φ, de la Z80, se obține o viteză maximă de 500 Kocteți/s.

BIBLIOGRAFIE

1. * * * Intel, *Component Data Catalog*, 1980.
2. * * * Intel, *Microprocessor and Peripherals Handbook*, 1983.
3. * * * CDC, *Product Specification for Flexible Disk Drive, Model 9404.*
4. * * * Hungarian Optical Works (MOM—Budapest, *Floppy Disk Storage Mechanism with Double Recording Density, Type MF 6400, EC-5082.*
5. * * * ISO 5654, *Information processing — data interchange on 200 mm (8 in) flexible disk cartridges using two-frequency recording at 13 262 ftprad on one side. Part 1: Dimensional, physical and magnetic characteristics. Part 2: Track format.*
6. * * * ISO 7065, *Information Processing — Data Interchange on 200 mm (8 in) Flexible Disk Cartridge Using Modified Frequency Modulation Recording at 13 262 ftprad on Two Sides. Part 1: Dimensional, Physical and Magnetic Characteristics. Part 2: Track Format for 1,9 tpmm (48 tpi).*
7. * * * Western Digital, *FD 179X Application Notes*, June, 1980.
8. GARTH, N., *Phase Locked Loop Design Fundamentals otorela* Application Note AN-535.
9. * * * NEC, *Considerations for Interfacing the NEC 765 to the CDC 9404 and 9406, -2, -3 Flexible Disk Drives*, Application Note 10.
10. FRANCHINI, R.C.; WARTNER, D. L., *A Method of High Density Recording on Flexible Magnetic Disks*, Computer Design, Octobcr 1976.
11. KALSTROM, D. J., *Simple Encoding Schemes Double Capacity of a Flexible Disk*, Computer Design, Septembcr 1976.
12. WORDEN, J., *Design Considerations for Dual-density Diskette Controllers*, Computer Design, June, 1978.
13. WEINER, R.; YORK, G., *A Single/Double Density Floppy Disk Controller Using the PD 765*, NEC Application Note 8.
14. GEBER, T.; VUICI, M.; CONSTANTINESCU, T.; NISIPEANU, M.; POPESCU, G.; BĂLEANU, C., *Echipamente periferice*, Editura Tehnică, București, 1981.
15. STĂNCESCU, ȘT., *Optimizarea utilizării codurilor în memoriile sistemelor de calcul*, Teză de Doctorat în Electronică Aplicată, I.P. București, 1985.
16. * * * MOTOROLA, *M6800 Application Manual*, 1975.
17. HOAGLAND, A. S., *Digital Magnetic Recording*, Willey, N. Y., 1963.

3

UN SISTEM MODULAR CU MICROPROCESOR

3.1. PREZENTAREA GENERALĂ A SISTEMULUI

Pentru a ilustra unele moduri de utilizare ale circuitelor din familia Z80 și ale circuitelor specializate de cuplare cu unitățile de discuri flexibile, vom prezenta în acest capitol elementele principale de proiectare a unui microcalculator organizat modular pe baza unei magistrale de sistem. Este cunoscut faptul că abordarea aplicațiilor cu microcalculatoare se poate face urmînd două direcții principale: prin utilizarea de elemente modulare la nivel de placă de circuit imprimat sau folosind microcalculatoare complet integrate. Prima soluție oferă mai multă flexibilitate și potențial de dezvoltare fiind adecvată, după părerea noastră, aplicațiilor care impun adaptări complexe de natură hardware. De asemenea, ea oferă posibilitatea organizării unei game largi de sisteme, pornind de la cele cu o complexitate foarte scăzută, de exemplu un microcalculator pe o singură placă, pînă la sisteme deosebit de complexe cum sînt sistemele multiprocesor. A doua soluție pare mai potrivită pentru aplicațiile ce necesită, în primul rînd, un efort deosebit de programare.

Concepția modulară a microcalculatoarelor, asupra căreia insistăm în mod implicit prin exemplul dat în acest capitol, este strîns legată de acțiunile de standardizare a magistralelor, elementelor constructive, software-ului. Ea permite utilizatorilor să-și concentreze activitatea în etapele de integrare a sistemelor: aplicațiile vor fi construite prin asamblarea mai multor produse modulare, compatibile cu o magistrală standard și coordonate printr-un sistem de operare, la care se adaugă modulele și programele specifice. Proiectarea aplicațiilor cu microprocesoare ar apărea în acest context mai rapidă depășindu-se prima etapă, aceea de implementare a modulelor de uz general: unități centrale, memorii, cuploare de discuri, interfețe serie/paralel, module de conversie analogic/numeric și numeric/analogic, surse etc.

Schema bloc a microcalculatorului este dată în figura 3.1. Este prezentată o structură minimă ce cuprinde un modul unitate centrală, UC, un modul memorie, MEM, și modulul pentru cuplarea unităților de discuri flexibile. Sistemul respectă tehnologia Eurocard, specificată de standardul 249-2 al Comisiei Electrotehnice Internaționale, utilizînd module dublu-Eurocard (160 ×233,4), figura 3.2, cu doi conectori cu 64 sau 96 de contacte, de tip

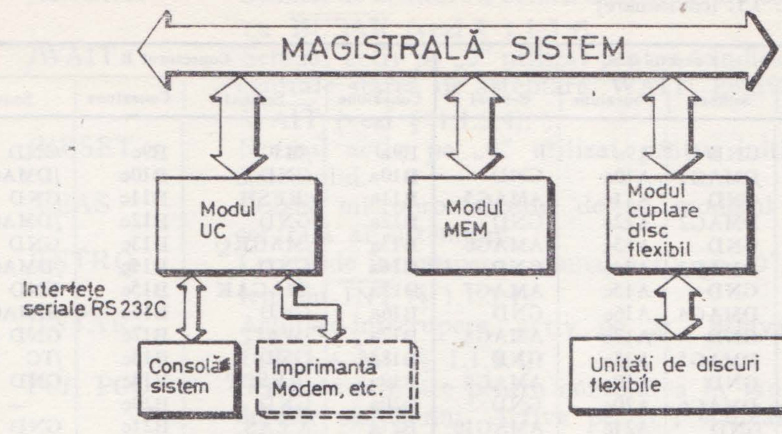


Fig. 3.1. Schema bloc a microcalculatorului

IEC 603-2 (fabricați și de întreprinderea CONECT). Extensia sistemului în funcție de aplicație se poate face prin adăugarea unor alte module compatibile cu modulele indicate în figura 3.1 prin intermediul magistralei de sistem, reprezentată fizic prin fundul de sertar. Această magistrală este o magistrală paralelă simplă, bazată pe semnalele microprocesorului Z80. Componența magistralei se prezintă în tabelul 3.1. Legăturile sistemului cu exteriorul se realizează prin intermediul panourilor atașate fiecărui modul, ca în figura 3.2, pe care se pot monta conectoare, comutatoare, LED-uri etc. În acest fel panourile au o dublă funcție asigurînd atît rigidizarea modulului în sertar, *rack*, cît și legătura lui cu exteriorul sistemului.

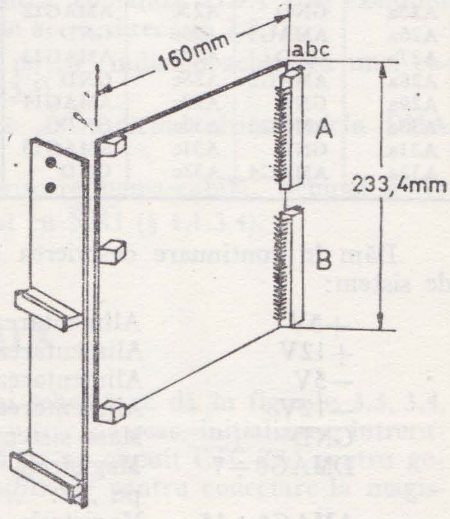


Fig. 3.2. Un modul dublu-Eurocard

TABELUL 3.1. Componența magistralei

Conectorul A				Conectorul B			
Conexiune	Semnal	Conexiune	Semnal	Conexiune	Semnal	Conexiune	Semnal
A1a	+5V	A1c	+5V	B1a	/MREQ	B1c	GND
A2a	+5V	A2c	+5V	B2a	GND	B2c	/DMARQ0
A3a	+5V	A3c	+5V	B3a	/IORQ	B3c	GND
A4a	+12V	A4c	+12V	B4a	GND	B4c	/DMARQ1
A5a	−5V	A5c	−5V	B5a	/RD	B5c	GND
A6a	−12V	A6c	−12V	B6a	GND	B6c	/DMARQ2
A7a	GND	A7c	—	B7a	/WR	B7c	GND
A8a	DMAG0	A8c	GND	B8a	GND	B8c	/DMARQ3

Conectorul A				Conectorul B			
Conexiune	Semnal	Conexiune	Semnal	Conexiune	Semnal	Conexiune	Semnal
A9a	GND	A9c	—	B9a	/M1	B9c	GND
A10a	DMAG1	A10c	GND	B10a	GND	B10c	/DMACK0
A11a	GND	A11c	AMAG5	B11a	/RFSH	B11c	GND
A12a	DMAG2	A12c	GND	B12a	GND	B12c	/DMACK1
A13a	GND	A13c	AMAG6	B13a	/MAGRQ	B13c	GND
A14a	DMAG3	A14c	GND	B14a	GND	B14c	/DMACK2
A15a	GND	A15c	AMAG7	B15a	/MAGAK	B15c	GND
A16a	DMAG4	A16c	GND	B16a	GND	B16c	/DMACK3
A17a	GND	A17c	AMAG8	B17a	/WAIT	B17c	GND
A18a	DMAG5	A18c	GND	B18a	GND	B18c	/TC
A19a	GND	A19c	AMAG9	B19a	/RESET	B19c	GND
A20a	DMAG6	A20c	GND	B20a	GND	B20c	—
A21a	GND	A21c	AMAG10	B21a	/CEAS	B21c	GND
A22a	DMAG7	A22c	GND	B22a	GND	B22c	—
A23a	GND	A23c	AMAG11	B23a	/INTRQ	B23c	GND
A24a	AMAG0	A24c	GND	B24a	GND	B24c	—
A25a	GND	A25c	AMAG12	B25a	/INTAK	B25c	GND
A26a	AMAG1	A26c	GND	B26a	GND	B26c	—
A27a	GND	A27c	AMAG13	B27a	PCI	B27c	GND
A28a	AMAG2	A28c	GND	B28a	GND	B28c	/NMI
A29a	GND	A29c	AMAG14	B29a	PCO	B29c	GND
A30a	AMAG3	A30c	GND	B30a	GND	B30c	GND
A31a	GND	A31c	AMAG15	B31a	GND	B31c	GND
A32a	AMAG4	A32c	GND	B32a	GND	B32c	GND

Dăm în continuare descrierea semnalelor din componența magistralei de sistem:

+5V	Alimentarea sistemului cu tensiune +5Vcc.
+12V	Alimentarea sistemului cu tensiune +12Vcc.
−5V	Alimentarea sistemului cu tensiune −5Vcc.
−12V	Alimentarea sistemului cu tensiune −12Vcc.
GND	Masa sistemului („0" logic).
DMAG0÷7	Magistrala de date. Intrări/ieșiri trei-stări active pe „1".
AMAG0÷15	Magistrala de adrese. Ieșiri trei-stări active pe „1". Asigură adresarea unui spațiu de memorie de maximum 64 Kocteți și a echipamentelor de I/E. De asemenea se poate utiliza pentru generarea adreselor de reîmprospătare la memoria dinamică.
/MREQ /IORQ /RD /WR /M1 /RFSH	Semnale de comandă (ieșiri trei-stări) echivalente cu $\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$, $\overline{WR}$, $\overline{M1}$, $\overline{RFSH}$ ale microprocesorului Z80 (vezi § 1.1.3.3.).
/MAGRQ	Semnal utilizat pentru a solicita modulului unitate centrală al sistemului magistralele de date, adrese și comenzi. Echivalent cu $\overline{BUSRQ}$ (vezi § 1.1.3.4).

/MAGAK	Semnal de achitare a cererii de magistrală echivalent cu $\overline{\text{BUSAK}}$ (vezi § 1.1.3.4).
/WAIT	Semnal activ pe „0" utilizat pentru a indica unității centrale starea de așteptare, WAIT. Echivalent cu $\overline{\text{WAIT}}$ (vezi § 1.1.3.4).
/RESET	Semnal activ pe „0" utilizat pentru inițializarea sistemului.
/CEAS	Ceasul microprocesorului de pe modulul unitate centrală al sistemului.
/INTRQ	Cerere de întrerupere; semnal activ pe „0" echivalent cu $\overline{\text{INT}}$ (§ 1.1.3.4).
/INTAK	Achitare-întrerupere, activ pe „0", echivalent cu $\overline{\text{IORQ}} + \overline{\text{M1}}$ (§ 1.1.3.4).
PCI, PCO	Semnale utilizate pentru conectarea în lanț a modulelor sistemului. Active pe „1", echivalente cu IEI, IEO (§ 1.2.2.3).
/DMARQ0÷3	Semnale active pe „0" utilizate pentru a solicita unui circuit pentru comanda DMA (de exemplu 8257) un ciclu de acces direct (§ 2.5.1).
/DMACK0÷3	Semnale active pe „0" indicînd achitarea unei cereri DMA (§ 2.5.1).
/TC	Semnal activ pe „0" indicînd ultimul ciclu DMA (§ 2.5.1).
/NMI	Cerere de întrerupere nemascabilă. Semnal activ pe „0" echivalent cu $\overline{\text{NMI}}$ (§ 1.1.3.4).

3.2. MODULUL UNITATE CENTRALĂ

Modulul *unitate centrală*, UC, a cărui schemă se dă în figurile 3.3, 3.4, 3.5, conține microprocesorul UC-Z80, circuitele de ceas, inițializare, întrerupere-panou, un circuit SIO-Z80 împreună cu un circuit CTC-Z80 pentru generarea frecvențelor de teletransmisie, *buffer*-ele pentru conectare la magistrala sistemului.

Circuitul de ceas este alcătuit dintr-un oscilator cu cuarț, a cărui frecvență este aleasă pentru a se obține după divizare cu doi și, apoi, cu ajutorul CTC-ului, frecvențe de teletransmisie cît mai apropiate de cele dorite (vezi tabelul 3.2), precum și un circuit de comandă specific realizat cu ajutorul unui tranzistor de comutație, aici 2N2907. Acest circuit de comandă, deși mărește prețul de cost, asigură funcționarea microprocesorului la frecvența de ceas maximă specificată de catalog, în cazul nostru 2,5 MHz, prin micșorarea frontului crescător al semnalului Φ. Pentru aplicații mici, care nu impun o funcționare a microprocesorului la frecvența maximă, intrarea de ceas poate fi comandată cu o poartă TTL avînd o rezistență de 330 Ω conectată la +5V. Fronturile pozitive ale ceasului cresc în acest caz la 60—80 ns, în funcție și de încărcarea liniei, ceea ce, evident, va conduce la micșorarea frecvenței maxime de lucru a sistemului.

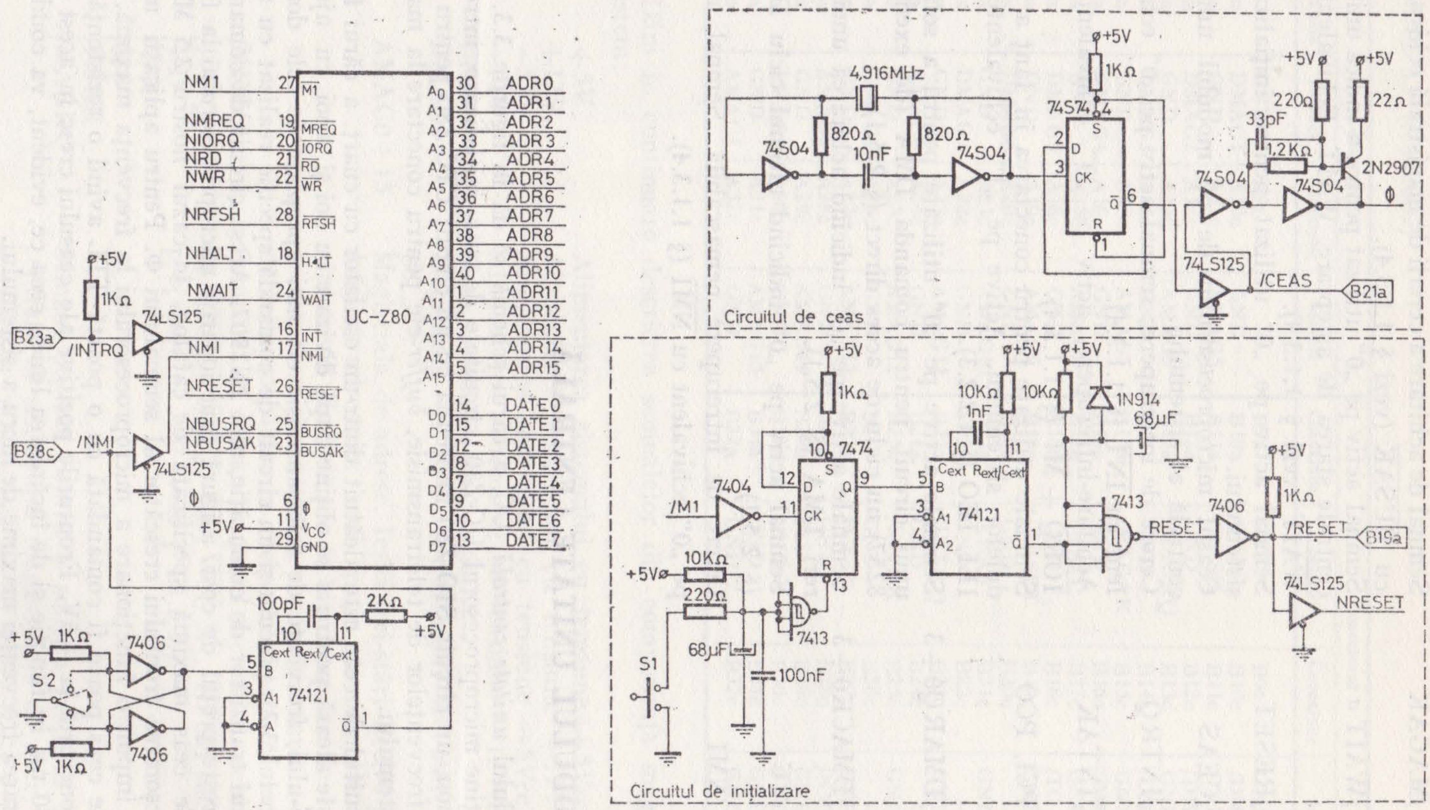


Fig. 3.3. Modulul UC. UC-Z80, inițializare, ceas, întrerupere-panou

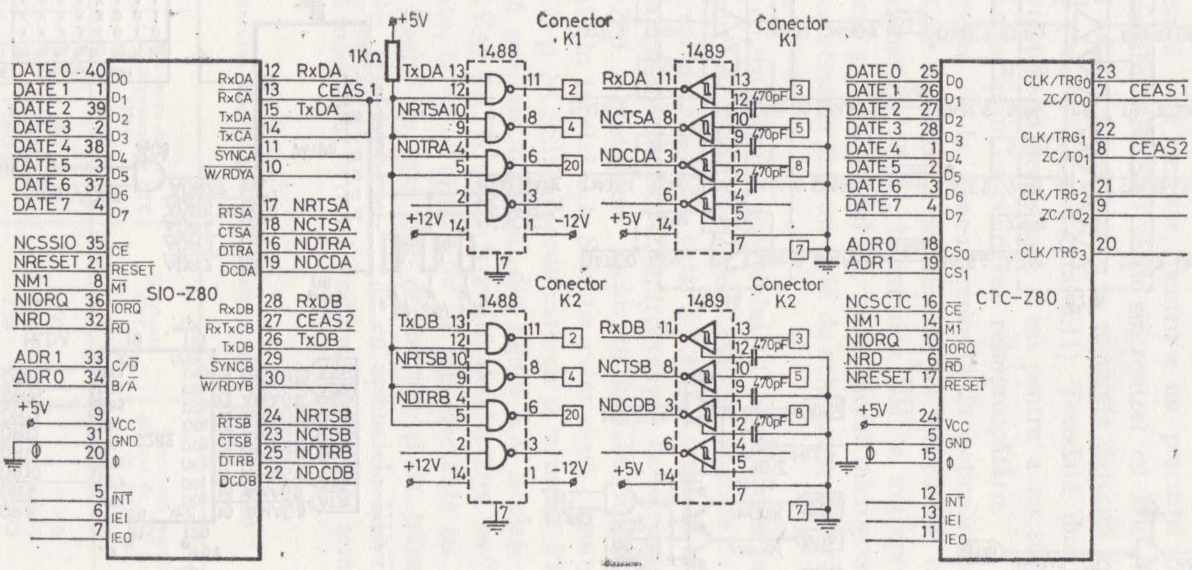
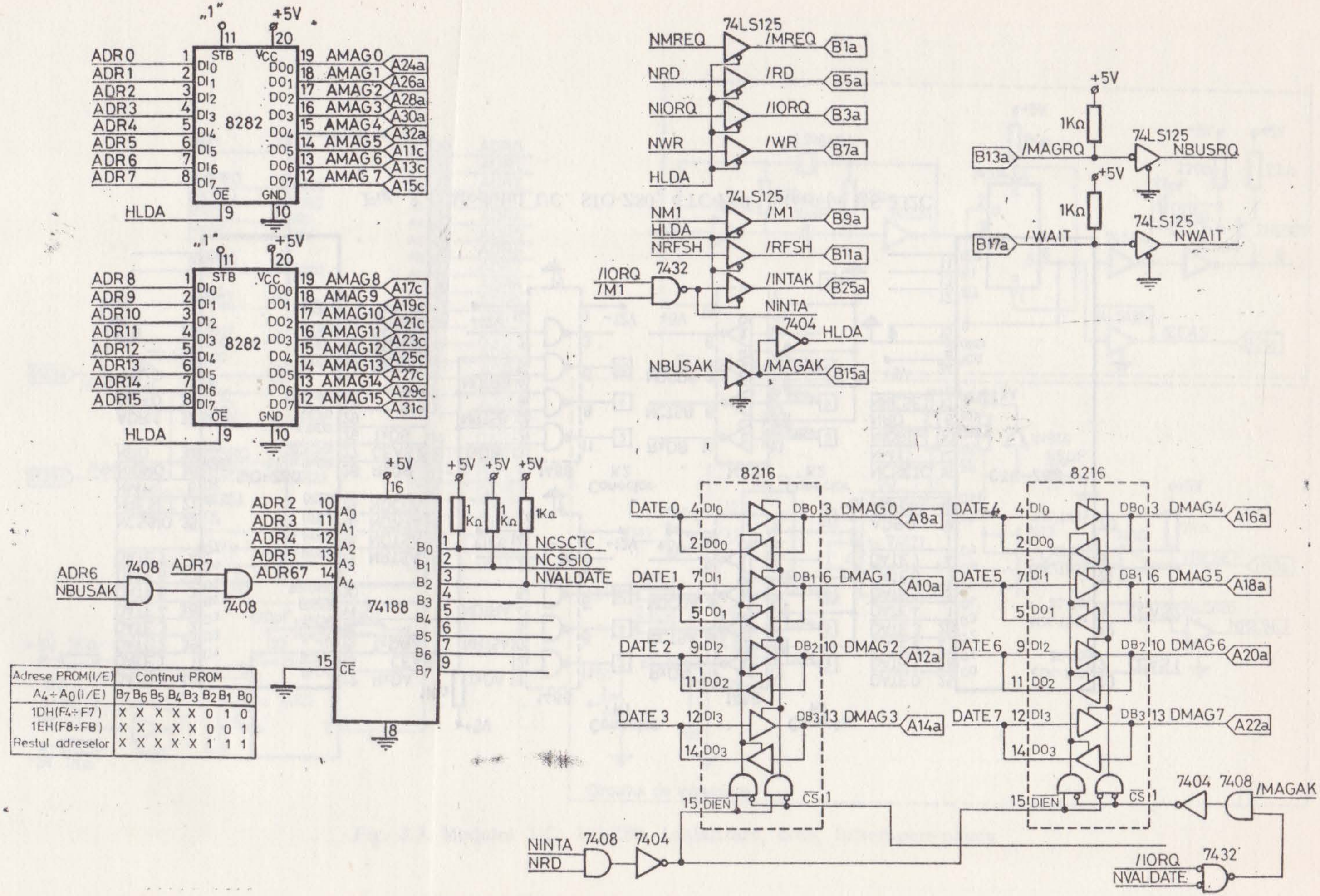


Fig. 3.4. Modulul UC. SIO-Z80, CTC-Z80, interfețe RS-232C



Adrese PROM(I/E) A4÷A0(I/E)	Conținut PROM B7 B6 B5 B4 B3 B2 B1 B0
1DH(F4÷F7)	X X X X X 0 1 0
1EH(F8÷FB)	X X X X X 0 0 1
Restul adreselor	X X X X X 1 1 1

Fig- 3-5, Modulul UC. Selecții, *buffer*-e de magistrală

Circuitul de inițializare realizează funcțiile de inițializare la punerea sub tensiune sau de la panou prin intermediul comutatorului S1. Așa cum am arătat în § 1.1.4, semnalul de inițializare a microprocesorului, NRESET, trebuie să fie activ cel puțin trei stări T. De asemenea, dacă sistemul are memorie dinamică este necesar, pentru a nu perturba conținutul acestei memorii, ca semnalul NRESET să fie sincronizat cu $\overline{M1}$, /M1 în figura 3.3, ceea ce inhibă acțiuni necontrolate la memorie datorită generării incorecte a lui NMREQ (vezi § 3.3 și figura 3.11). Totodată durata impulsului de inițializare nu trebuie să depășească 1 ms pentru a nu se pierde conținutul memoriei dinamice prin suspendarea reimprospătării.

Întreruperea de la panou, generată la acționarea comutatorului S2. este memorată de un *latch* ce declanșează un monostabil inițiind un impuls scurt, de aproximativ 150 ns, la intrarea $\overline{NMI}$ a microprocesorului. Acest impuls, memorat și el, la rîndul lui, în interiorul microprocesorului, va conduce la trecerea acestuia într-un ciclu de achitare a întreruperii nemascabile (vezi § 1.1.4.).

Legătura cu exteriorul a modulului UC se face prin intermediul a două conectoare de tip RK 25 (fabricate de CONECT) montate pe panou, asigurînd două interfețe seriale RS 232-C (vezi § 1.4.1). Comanda acestor interfețe se face cu ajutorul circuitului SIO-Z80, frecvențele de teletransmisie fiind generate cu CTC-Z80. Constantele încărcate în CTC în funcție de frecvența aleasă sînt date în tabelul 3.2. Selecția celor două circuite se face cu semnalele NCSCTC și NCSSIO, decodificate cu o memorie PROM din adresele ADR2 ÷ ADR6 conform tabelului din figura 3.5. Același PROM generează și semnalul NVALDATE utilizat pentru deselectarea *buffer*-elor de pe magistrala de date, în cazul operațiilor de I/E efectuate cu SIO sau CTC din modulul UC. *Buffer*-ele mai sînt deselectate și în transferurile DMA, cu /MAGAK activ. Direcția se comandă cu semnalele NINTA și NRD: citire spre interiorul modulului în cazul cînd unul din aceste semnale devine activ,

TABEL 3.2. **Generarea frecvențelor de teletransmisie cu CTC-Z80**

Frecvență teletransmisie (în baud)	Frecvența dorită a ceasului generat de CTC [Hz]	Constantă CTC	Frecvența ceasului generat de CTC cu Φ de 2,458 MHz [Hz]	Eroare
110	1 760	1 392	1 766	+0,34%
300	4 800	512	4 801	+0,02%
600	9 600	256	9 602	+0,02%
1 200	19 200	128	19 203	+0,02%
2 400	38 400	64	38 406	+0,02%
4 800	76 800	32	76 813	+0,02%
9 600	153 600	16	153 625	+0,02%

3.3. MODULUL MEMORIE

Schema modulului de memorie al microcalculatorului prezentat se dă în figura 3.6. Modulul conţine două circuite EPROM şi maximum 64 Kocteţi de memorie RAM dinamică organizată în patru rînduri de cîte 8 capsule de 16K × 1 biţi, de tipul 4116. Cele două EPROM-uri conţin un program de încărcare a conţinutului lor în memoria RAM, programul MONITOR şi programele de tratare a întreruperilor de disc flexibil. La punerea sub tensiune, după iniţializare, bistabilul START se poziţionează pe „1" validînd citirea de la adresa 0000H a memoriilor EPROM şi scrierea lor în memoria RAM. După transcrierea programelor din EPROM în RAM, se dă controlul unui program de testare a stării discului în funcţie de care se va trece în MONITOR sau se va încărca sistemul de operare. Totodată saltul la programul de testare, aflat în a doua jumătate a spaţiului de memorie, va forţa bistabilul START pe „0", invalidînd memoriile EPROM, permiţînd în acelaşi timp şi citirea din RAM. Astfel sistemul poate avea de fapt la dispoziţie maximum 60K RAM, 4 K fiind ocupaţi cu conţinutul celor două EPROM-uri. Organizarea memoriei sistemului precum şi detaliile de punere în funcţiune a lui se vor da în § 3.5.3.

Pentru reîmprospătarea memoriilor dinamice se cunosc două soluţii, [6]: una sincronă în care generarea semnalului $\overline{CAS}$ se face cu ajutorul unui bistabil şi alta asincronă, ce utilizează o linie de întîrziere. În figura 3.6 se dă o variantă a primei metode.

După cum am menţionat, UC-Z80, are avantajul, faţă de alte microprocesoare pe 8 biţi, de a permite ataşarea uşoară a memoriilor dinamice. Semnalele de comandă asociate operaţiilor cu memoria dinamică, citire, scriere sau reîmprospătare, sînt $\overline{MREQ}$, $\overline{RD}$ $\overline{WR}$ şi $\overline{RFSH}$. Semnificaţia acestor semnale a fost dată în § 1.1.3. În figurile 3.7, 3.8, 3.9 reluăm diagramele de timp care se referă la aceste semnale de comandă, precizînd întîrzierile şi relaţiile dintre ele. Perioada ceasului, $t_c = 406$ ns, corespunde frecvenţei cuarţului de 4,916 MHz. Timpii de acces la memorie impuşi pentru o funcţionare corectă în cazul unui ciclu de extragere şi al unui ciclu de citire-memorie se pot calcula cu relaţiile:

$$t_{\text{acces cod-op}} = 3 \cdot \frac{t_c}{2} - t_{DL\bar{\Phi}(MR)} - t_{S\Phi(D)} \qquad (3.1)$$

$$t_{\text{acces citire}} = 4 \cdot \frac{t_c}{2} - t_{DL\bar{\Phi}(MR)} - t_{S\bar{\Phi}(D)} \qquad (3.2)$$

Cu $t_c = 406$ ns, $t_{DL\bar{\Phi}(MR)} =$ max 100 ns, $t_{S\Phi(D)} =$ min 50 ns şi $t_{S\bar{\Phi}(D)} =$ = min 60 ns rezultă:

$$t_{\text{acces cod-op}} = 459 \text{ ns}$$

$$t_{\text{acces citire}} = 652 \text{ ns.}$$

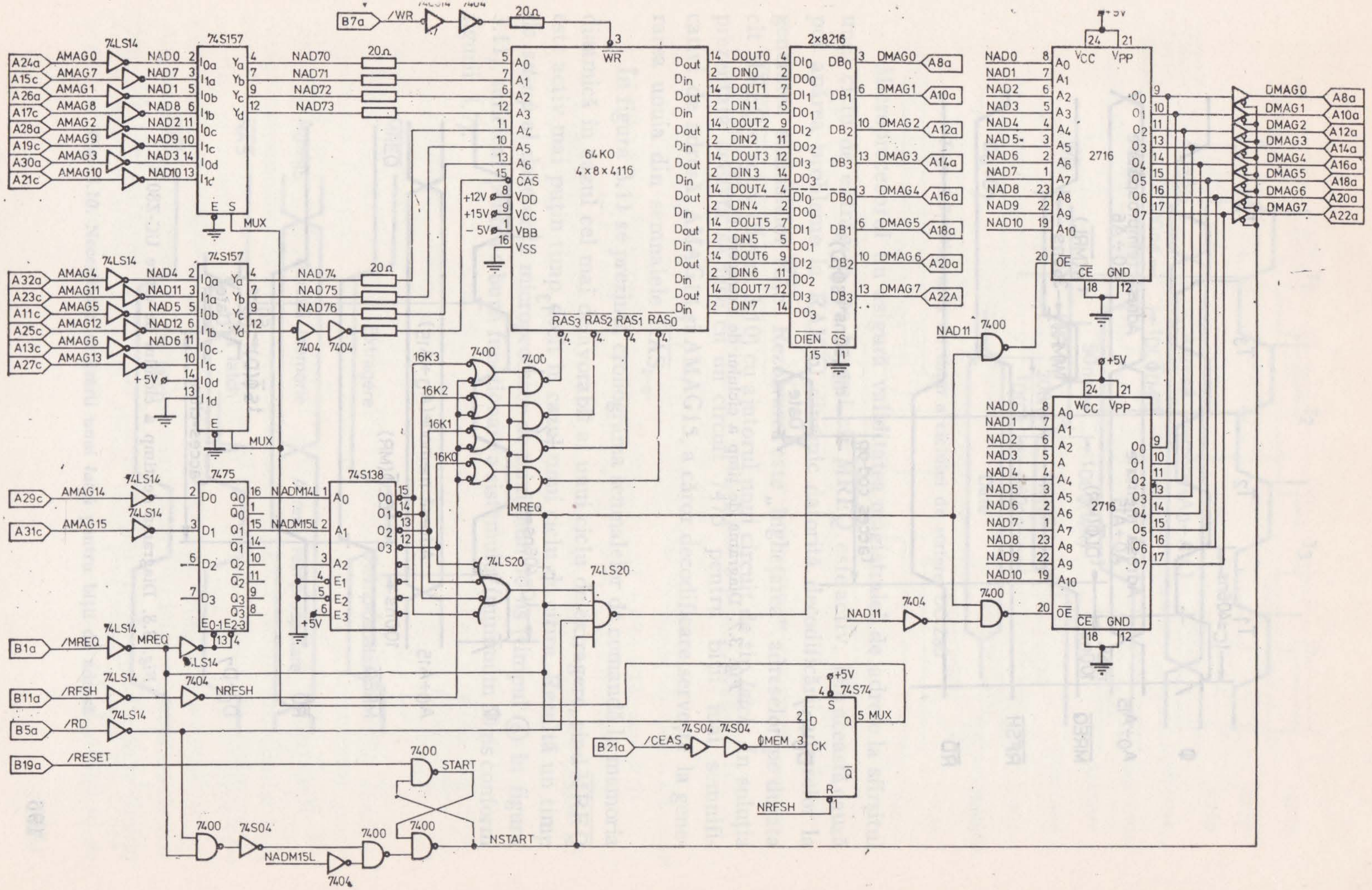


Fig. 3.6. Schema modulului de memorie

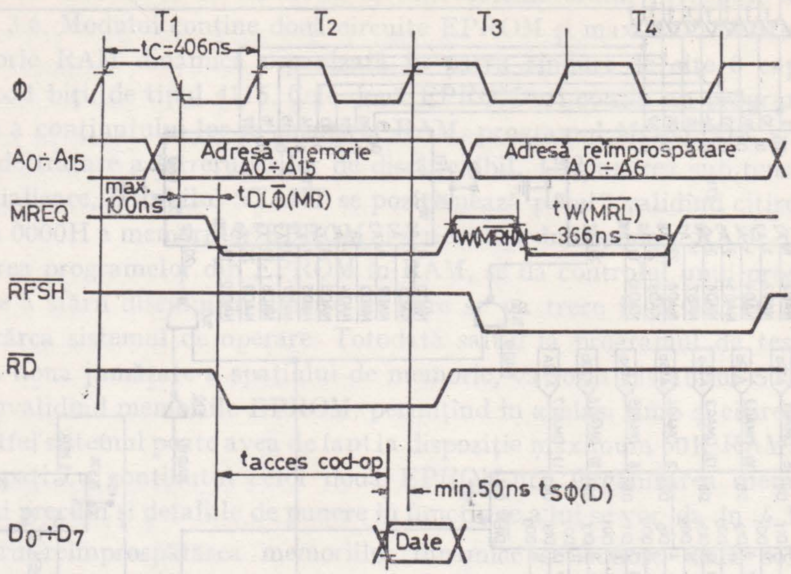


Fig. 3.7. Diagrama de timp a ciclului de extragere UC-Z80

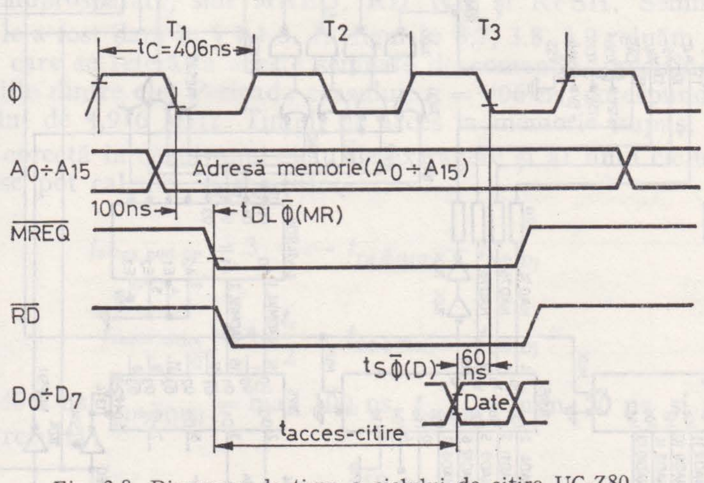


Fig. 3.8. Diagrama de timp a ciclului de citire UC-Z80

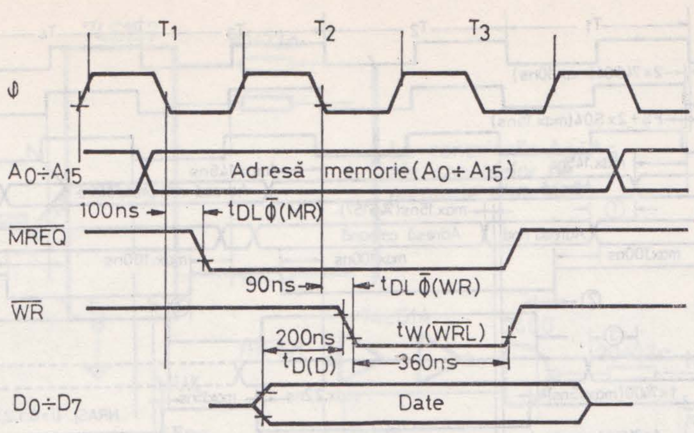


Fig. 3.9. Diagrama de timp a ciclului de scriere UC-Z80

Microprocesorul nu asigură validitatea magistralei de adrese la sfîrșitul unui ciclu de extragere, pe timpul cît $\overline{MREQ}$ este activ. Din această cauză pot apărea probleme la RAM-ul dinamic datorită decodificării adreselor la generarea semnalelor $\overline{RAS}$. Rezolvarea este „înghețarea" adreselor pe durata cît $\overline{MREQ}$ e activ (figura 3.10) cu ajutorul unui circuit de tip *latch*. În soluția prezentată aceasta se face cu un circuit 7475 pentru biții mai semnificativi de adresă, AMAG14 și AMAG15, a căror decodificare servește la generarea unuia din semnalele $\overline{RAS}_{3-0}$.

În figura 3.11 se prezintă cronograma semnalelor de comandă la memoria dinamică în cazul cel mai defavorabil al unui ciclu de extragere, cînd $\overline{MREQ}$ este activ mai puțin timp decît în cazul unui ciclu de citire. Rezultă un timp de *set-up* al datelor la microprocesor de minimum 70 ns (timpul ⑥ in figura 3.11), suficient pentru o bună funcționare a sistemului (minimum 50 ns conform figurii 3.7).

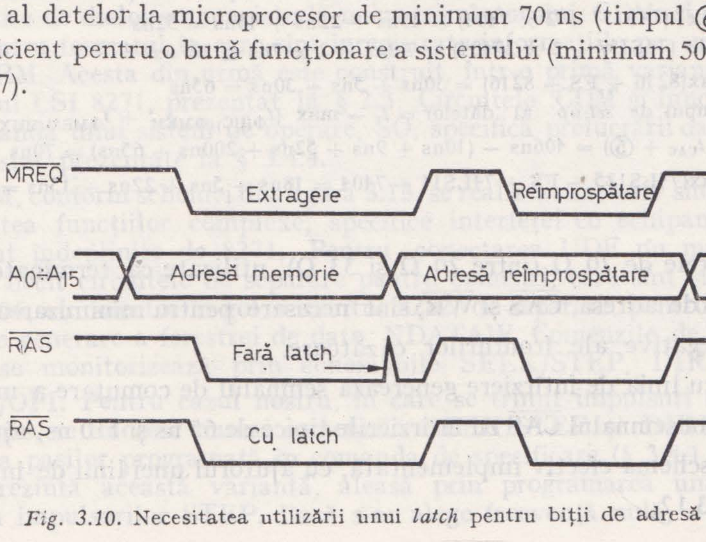


Fig. 3.10. Necesitatea utilizării unui *latch* pentru biții de adresă

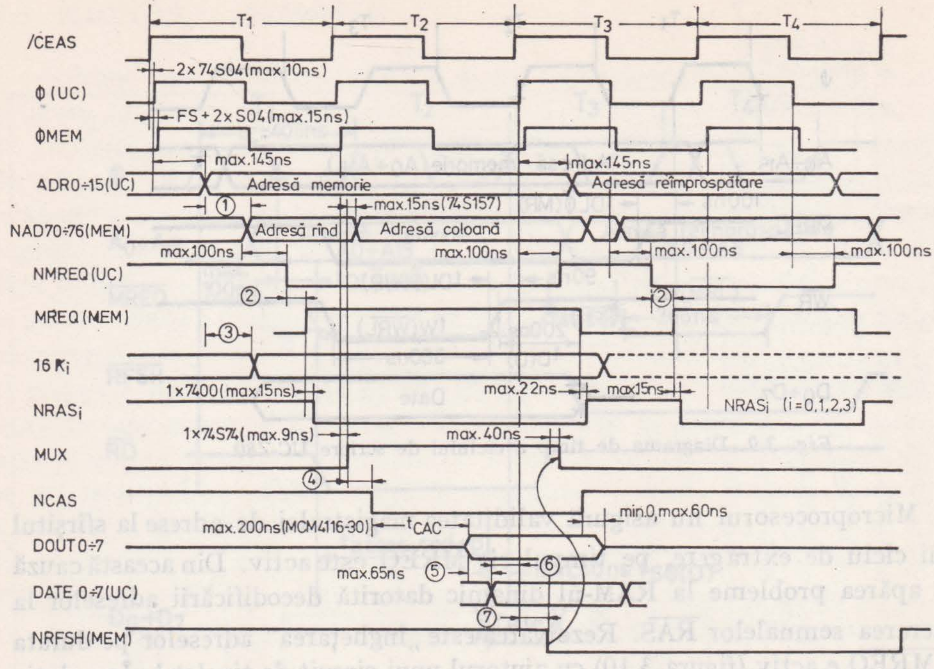


Fig. 3.11. Diagrama de timp a semnalelor de comandă la memoria dinamică în cazul unui ciclu de extragere

Notă: FS = fund de sertar

① = max (8282+FS + 74LS14 + 74S157) = 30ns + 5ns + 22ns + 7ns = 64ns

② = max (74LS125 + FS + 74LS14) = 18ns + 5ns + 22ns = 45ns

③ = max (8282 + FS + 74LS14 + 7475 + 74LS138) = 30ns + 5ns + 22ns + + 30ns + 12ns = 99ns

④ = max (74S157 + 2 × 7404) = 15ns + 22ns + 15ns = 52ns
tipic (74S157 + 2 × 27404) = 7,4ns + 2 × 7ns ≅ 22 ns

⑤ = max(8216 + FS + 8216) = 30ns + 5ns + 30ns = 65ns

⑥ = timpul de *set-up* al datelor = t_c − max ($t_{\Phi(UC)/\Phi MEM}$ + $t_{\Phi MEM/MUX}$ + ④ + + t_{CAC} + ⑤) = 406ns − (10ns + 9ns + 52ns + 200ns + 65ns) = 70ns

⑦ = max(74LS125 + FS + 74LS14 + 7404 = 18ns + 5ns + 22ns + 15ns = 60ns

Rezistențele de 20 Ω (între 20 Ω și 33 Ω), utilizate ca terminatori-serie pentru liniile de adresă, $\overline{CAS}$ și $\overline{WR}$, sînt necesare pentru minimizarea supracreșterilor negative ale fronturilor căzătoare.

Soluția cu linia de întîrziere generează semnalul de comutare a multiplexoarelor și apoi semnalul $\overline{CAS}$ cu întîrzierile tipice de 65 ns și 110 ns, ajustabile funcție și de schema efectiv implementată, cu ajutorul unei linii de întîrziere, ca în figura 3.12.

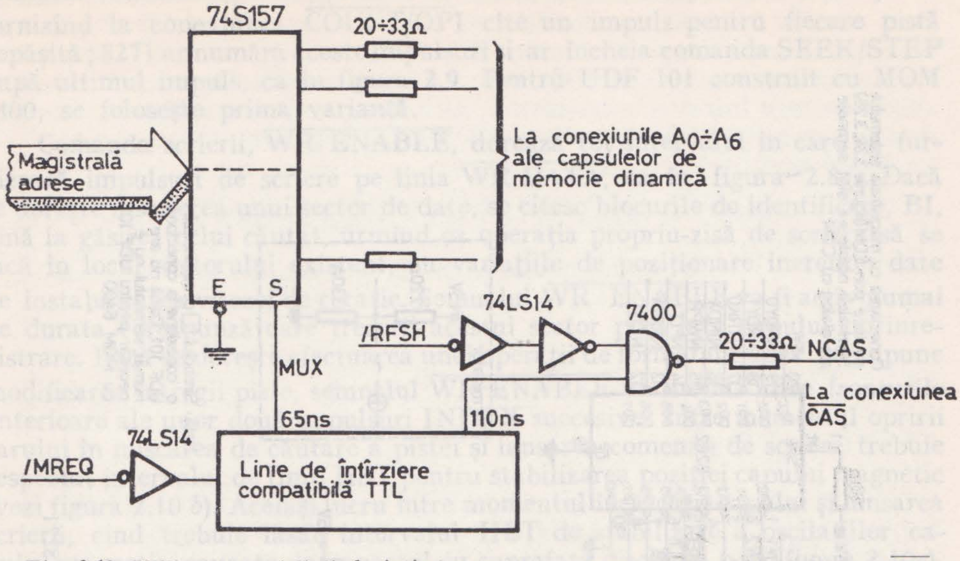


Fig. 3.12. Utilizarea unei linii de întîrziere pentru generarea semnalelor MUX și $\overline{CAS}$

3.4. MODULE DE CUPLARE A DISCULUI FLEXIBIL

3.4.1. CUPLOR DE DISC FLEXIBIL CU LSI 8271

3.4.1.1. PREZENTAREA CIRCUITELOR

Subsistemul de disc flexibil SSDF este compus dintr-o unitate duală de disc flexibil UDF și un cuplor de disc flexibil CDF, alcătuit, la rîndul său, din circuite de dialog cu magistrala microcalculatorului, CDM, și circuite specifice pentru formatul în care sînt înregistrate informațiile pe suportul magnetic, FRM. Acesta din urmă este construit, într-o primă variantă, în jurul circuitului LSI 8271, prezentat la § 2.3. Circuitele CDM și funcționarea lor sub comanda unui sistem de operare, SO, specifică prelucrării datelor cu disc flexibil, sînt prezentate la § 3.4.4.

FRM, conform schemei din figura 3.13, se realizează relativ simplu, fiindcă majoritatea funcțiilor complexe, specifice interfeței cu echipamentul periferic, sînt îndeplinite de 8271. Pentru conectarea UDF nu mai rămîn de adăugat decît circuitele de separare pentru comenzi, care sînt simple inversoare 7406, cu colector-în-gol, circuitul 8282, de separare, pentru stări și circuitul de generare a ferestrei de date, NDATAW. Comenzile de deplasare a carului se monitorizează prin conexiunile SEEK/STEP, DIRECTION și $\overline{\text{COUNT}}$/OPI. Pentru cazul nostru, în care se trimit impulsuri de comandă pas cu pas, se folosesc numai comenzile SEEK/STEP și DIRECTION, cu frecvența pașilor programată în comanda de specificare (§ 3.4.1.2.1). Figura 2.10 *a* prezintă această variantă, aleasă prin programarea unei frecvențe nenule a impulsurilor STEP. Dacă s-ar alege frecvență nulă, s-ar trimite o

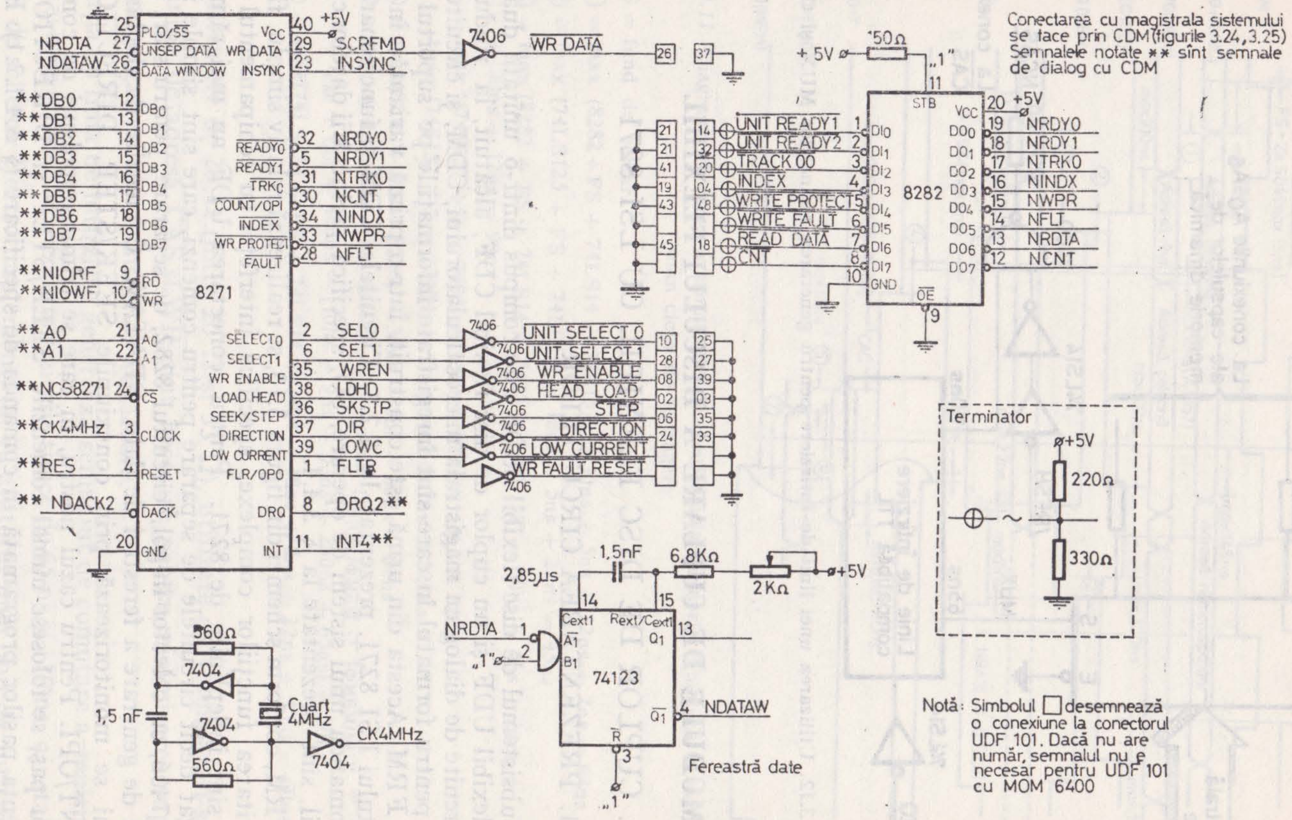


Fig. 3.13. Cuplor de disc flexibil cu 8271

comandă continuă de SEEK/STEP, iar UDF ar efectua deplasarea fără oprire, furnizînd la conexiunea $\overline{\text{COUNT/OPI}}$ cîte un impuls pentru fiecare pistă depășită ; 8271 ar număra aceste impulsuri și ar încheia comanda SEEK/STEP după ultimul impuls, ca în figura 2.9. Pentru UDF 101 construit cu MOM 6400, se folosește prima variantă.

Comanda scrierii, $\overline{\text{WR ENABLE}}$, durează tot intervalul în care se furnizează impulsuri de scriere pe linia $\overline{\text{WR DATA}}$, ca în figura 2.8*c*. Dacă se dorește înscrierea unui sector de date, se citesc blocurile de identificare, BI, pînă la găsirea celui căutat, urmînd ca operația propriu-zisă de scriere să se facă în locul sectorului existent, cu variațiile de poziționare inerente, date de instabilitatea vitezei de rotație. Semnalul $\overline{\text{WR ENABLE}}$ va fi activ numai pe durata corespunzătoare trecerii acestui sector prin fața capului de înregistrare. Dacă se dorește efectuarea unei operații de formatare, care presupune modificarea întregii piste, semnalul $\overline{\text{WR ENABLE}}$ va fi emis între fronturile anterioare ale unor două impulsuri INDEX succesive. Între momentul opririi carului în mișcarea de căutare a pistei și lansarea comenzii de scriere trebuie respectat intervalul de timp HST pentru stabilizarea poziției capului magnetic (vezi figura 2.10 *b*). Același lucru între momentul încărcării capului și lansarea scrierii, cînd trebuie lăsat intervalul HLT de stabilizare a oscilațiilor capului magnetic, cauzate de impactul cu suprafața dischetei (vezi figura 2.10*c*). HST și HLT se programează în comanda de specificare, în funcție de caracteristicile UDF.

Semnalul de scriere $\overline{\text{WR DATA}}$ corespunde formatului FM (§ 2.2.1), cu celulă-bit de 4 μs, în care simbolul de la început este prezent întotdeauna, pentru sincronizare, iar simbolul din centru apare la înregistrarea informației „1". Dacă se înregistrează „0" ,impulsul central lipsește. Pentru minidisc duratele se dublează, celula ajungînd la 8 μs. Semnalul citit, transmis pe interfața UDF, corespunde semnalului scris, cu aceleași durate. Evident acest semnal, $\overline{\text{UNSEP DATA}}$, are variații în timp, datorate modificărilor produse de viteza de rotație, de interferența intersimbol, de alunecarea magnetică etc. Din această cauză, circuitul 8271 are nevoie să primească, pe lîngă semnalul cules de pe capul magnetic amplificat și format, $\overline{\text{UNSEP DATA}}$, un alt semnal, derivat din acesta, care să corespundă localizării impulsurilor de informație. Dacă se folosește opțiunea cu monostabil, cu conexiunea externă PLO/$\overline{\text{SS}}$ la masă, figura 2.7*b*, se generează $\overline{\text{DATA WINDOW}}$ la ieșirea $\overline{Q}$ a circuitului 74123. Elementele pasive ale monostabilului redeclanșabil se aleg pentru o durată care să depășească impulsul de date din centrul celulei-bit. Acesta va fi eliminat din șirul $\overline{\text{UNSEP DATA}}$, obținîndu-se semnalul de sincronizare dorit. La aplicarea soluției cu buclă cu calare pe fază, PLL, conexiunea PLO/$\overline{\text{SS}}$ la „1", ca în figura 2.7*a*, circuitele exterioare generează un semnal de sincronizare mediat pe secvența precedentă, pentru fiecare celulă-bit. Avantajul utilizării unei PLL derivă din factorul de umplere constant al celulei-bit, indiferent de variațiile de viteză ale dischetei și din toleranța la lipsa accidentală a impulsurilor de sincronizare, care pot fi refăcute din șirul de impulsuri anterioare. O variantă de circuit PLL, cu amplificator operațional μA741 în comparatorul de fază și cu oscilator comandat în tensiune 74S124 se va prezenta în § 3.4.2.

Pentru adaptarea la magistrala microcalculatorului, modulul are prevăzute două circuite 8216 pentru amplificarea atacului liniilor DMAG0÷7 și amplificatoare de separare pentru liniile /RD, /WR, /IORQ și /RESET, figura 3.23. Oscilatorul cu cuarț de 4 MHz figura 3.13 se face cu inversoarele 7404 ca amplificatoare într-o schemă clasică. Selecția circuitului 8271, este generată dintr-un circuit FPLA, din schema CDM ce va fi prezentată la § 3.4.4. Ecuația logică pentru NCS8271 va fi:

$$NCS8271 = (F0 + F1 + F2) \cdot \overline{NIORQ} \cdot NAEN \cdot NRES$$

unde F0, F1, F2, corespund mintermilor cu combinațiile liniilor AMAG0÷7 pentru decodificarea selecției, în notație hexazecimală.

3.4.1.2. PROGRAMAREA LSI 8271 (Anexa A)

3.4.1.2.1. **Inițializarea**

Rutina de inițializare, I8271, respectă, în mare, organigrama din figura 2.15. Subrutina RESET, prima apelată, trimite impulsul de inițializare software în registrul RRES. Prin succesiunea operațiilor de înscriere - citire a registrului special de mod, se testează prezența pe magistrală a modulului CDF. Microcalculatorul este înștiințat dacă 8271 există sau nu în configurație, în funcție de coincidența octetului înscris cu cel citit. Operația de înscriere în registrul de mod, pe de altă parte, informează 8271 asupra prezenței circuitului DMA pe magistrală și asupra variantei constructive de UDF, cu unul sau două cărucioare. În exemplul dat, transferurile se fac cu DMA iar UDF duală este formată din două unități separate, avînd prin urmare două cărucioare (în registrul de mod se înscrie 0C0H). Următoarea subrutină stabilește vectorii de întrerupere, pregătind modul de lucru al CDM, expus în § 3.4.4. În continuare, subrutina SPECF programează 8271 cu parametrii corespunzători caracteristicilor UDF conectate. În cazul nostru, comanda de specificare va avea parametrii pentru MOM 6400, cu SRT = 4 ms, HST = 20 ms, NROT = = 6 rotații, HLT = 45 ms. Operația de specificare se încheie după faza de comandă, FC, și nu necesită faze de execuție FE sau rezultat, FR. După înscrierea numerelor a două piste defecte (0FFH la inițializare), se trece la recalibrarea și testarea stării READY a fiecărei UDF. Recalibrarea se realizează prin operație de căutare a pistei 0, iar starea liniei READY se află prin ordin de testare a stării UDF (cod 02CH), trimis de două ori pentru fiecare unitate. Repetarea este necesară pentru anularea memorării NOT READY, în urma inițializării 8271. Ordinul de citire stare UDF, ca și cel de citire a registrelor speciale, permite trecerea imediată din FC în FR, fără lansare de întrerupere, IT. Ordinul de specificare, ca și cel de înscriere a registrelor speciale, nu cere nici FR, execuția sa încheindu-se după faza FC. Operația de inițializare se consideră încheiată după actualizarea indicatorului cu stările READY/NOT READY ale UDF, pentru informarea microcalculatorului asupra configurației SSDF.

3.4.1.2.2. Programarea 8271 pentru execuția unei operații curente (cod < 02CH)

După un dialog, justificat în § 3.4.4., în vederea interacțiunii cu sistemul de operare, programul ajunge în faza in care operația ce va fi executată de 8271 este cunoscută (căutare, scriere, citire, formatare). Pentru operațiile care presupun transfer de date, se calculează întîi numărul de transferuri DMA necesare, în funcție de numărul de sectoare cerut, după care se programează circuitul 8257, în prealabil inițializat, cu adresa, numărul de octeți și sensul transferului. Se lucrează pe canalul 2, în varianta cu STOP pe *Terminal Count*, TC (cod 44H în registrul de comandă DMA).

Codul specific al operației este completat cu adresa UDF și trimis la 8271, dacă acesta nu este ocupat cu o operație în curs (test BSY, poziția 7 în RS). Testîndu-se apoi și poziția 5 în RS — RP plin — se furnizează ulterior și parametrii referitori la transfer. Subrutina COM trimite codul comenzii în RC, COMP trimite codul comenzii și un parametru, iar COMPP trimite codul comenzii și doi parametri. Parametri suplimentari se pot trimite separat, cu subrutina BP. După acceptarea ultimului parametru, faza FC se încheie și microprocesorul trece în HALT. Se așteaptă apariția unei întreruperi, care semnalează încheierea FE, normală sau anormală. Apariția întreruperii trece 8271 în FR, iar programul de tratare a întreruperii preia rezultatul, cu tipul și codul erorii, dacă aceasta a apărut. După revenirea în programul întrerupt, care a conținut instrucțiunea HALT, se reactualizează indicatorii de stare UDF pentru informarea microprocesorului asupra unor eventuale schimbări. Aceasta se realizează printr-o nouă interogare a UDF-urilor cu cîte o comandă de citire stare.

3.4.1.2.3. Tratarea întreruperii lansate de 8271 (subrutina I82714)

Faza de rezultat, FR, este parcursă în rutina de tratare a întreruperi lansate de 8271. Rutina face, în plus, conversie de cod a erorii pentru codul specific al sistemului de operare, SO. Analizîndu-se, pe rînd, din octetul rezultat, mai întîi tipul și apoi codul erorii, se obține o adresă în tabelul de 16 octeți în coduri, pe care le acceptă SO. Fiecare octet va avea semnificațiile de mai jos:

Octeți 0÷3, cod 00; fără eroare (operația SCAN nu este implementată).

Octet 4, cod 03; eroare de sincronizare, în care lipsește un impuls de „ceas" (se aplică numai în afara mărcilor așteptate la intervale validate de INSYNC). Operația s-a încheiat imediat, cu lansarea întreruperii.

Octet 5, cod 10H; eroare de ritm, în cazul în care DMA-ul nu conlucrează în timp util cu 8271. Operația este terminată imediat, cu lansarea de întrerupere.

Octet 6, cod 0AH; eroare CRC pe BI, cînd nu s-a găsit nici un BI cu CRC corect pentru numărul de sector cerut. Transferul nu a avut loc.

Octet 7, cod 02; eroare CRC pe sector, caz în care sectorul a fost identificat printr-un BI corespunzător, marca de sector a fost localizată și s-au transferat datele din sector, dar la verificarea CRC octeții de la sfîrșit nu corespund cu cei deduși din conținutul sectorului. Transferul în memorie a avut loc, dar datele transferate trebuie considerate incorecte.

Octet 8, cod 80H; unitatea adresată in stare NOT READY, avînd drept cauză: UDF neconectată, UDF nealimentată, dischetă neincărcată cu uşa închisă, UDF a trecut de la ultima operaţie prin starea NOT READY chiar dacă unitatea este pregătită în momentul prezentării noii comenzi. Starea NOT READY este memorată în 8271, de unde este ştearsă numai *după* trimiterea unei comenzi de citire a stării UDF.

Octet 9, cod 20H; se intenţionează scriere pe o dischetă protejată. Operaţia de scriere nu se mai produce, atît din cauză că UDF nu o permite, dar şi pentru că 8271 nu o va mai lansa spre unitate.

Octet 10, cod 04; eroare la recalibrare, cînd, după retragerea căruciorului cu pînă la 255 paşi, nu s-a activat semnalul $\overline{\text{TRACK 00}}$.

Octet 11, cod 40H; eroare la scriere, cînd UDF a semnalat $\overline{\text{WRITE FAULT}}$. În lipsa sa de pe interfaţă, semnalul trebuie menţinut inactiv.

Octeţii 12 ÷ 15, cod 0FH; lipsă a sectorului căutat, cînd 8271 nu a reuşit să identifice sectorul cerut şi au trecut două impulsuri $\overline{\text{INDEX}}$. Se poate lansa şi dacă numărul de pistă nu s-a verificat la citirea BI. În acest din urmă caz, 8271 încearcă automat să găsească sectorul cerut, pe încă două piste adiacente interioare.

3.4.2. CUPLOR DE DISC FLEXIBIL CU LSI 8272

3.4.2.1. PREZENTAREA CIRCUITELOR

Circuitul LSI 8272, echivalent cu μPD 765, oferă, faţă de circuitul LSI 8271, avantajul esenţial de a lucra în două moduri de înregistrare — densitate simplă (FM) sau densitate dublă (MFM). Cînd lucrează în densitate simplă, organizarea informaţiei este compatibilă cu cea oferită de 8271, în standard IBM 3740, ISO 5654. Dacă densitatea este dublă, informaţia devine conformă unui standard acceptat internaţional, ISO 7065, respectînd caracteristicile din echipamentul IBM System 34. Pentru dischetele standard de 8 *inch*, celula-bit de 4 μs la modul FM, se reduce la 2 μs în densitate dublă, mod MFM, fără schimbarea densităţii maxime de înregistrare a tranziţiilor, care vor rămîne în continuare depărtate la cel puţin 2 μs. Dacă la FM duratele între tranziţii sînt de 2 μs şi 4 μs, corespunzătoare înregistrărilor succesive de simboluri de informaţie „1", respectiv „0", la MFM există intervale de 2 μs, 3 μs şi 4 μs, în funcţie de secvenţa de cod. Mărimea densităţii de înregistrare impune rezolvarea unor probleme de sincronizare. Dacă la densitate simplă, FM, fiecare celulă asigură sincronizarea printr-un simbol de „ceas" plasat între simbolurile ce poartă informaţia de înregistrat, la densitate dublă, MFM, unele simboluri de „ceas" se elimină din secvenţa de cod şi poziţia lor probabilă trebuie dedusă din poziţiile tranziţiilor precedente. Circuitul care rezolvă sincronizarea în MFM este o buclă cu calare pe fază, PLL, şi se include în modulul CDF cu 8272. Circuitul PLL, utilizat şi în densitate simplă, contribuie la micşorarea numărului erorilor de lectură, în comparaţie cu varianta de cuplor cu LSI 8271 fără PLL. Pe de altă parte, celula de bit micşorată va mări în funcţie de context, valoarea alunecării relative a poziţiei reale a tranziţiei faţă de poziţia sa teoretică, astfel că se impune, în densitate dublă, aplicarea procedeului de *precompensare* în scriere, în vederea aducerii interferenţei inter-

simbol în limite tolerabile. 8272 furnizează semnale de control pentru alegerea valorii și sensului precompensării, dar sînt necesare circuite suplimentare, figura 3.15, pentru a realiza generarea secvențelor, decalate cu intervalul de precompensare. Modulul de CDF cu 8272 include, ca și modulul precedent, circuitele de dialog cu magistrala microcalculatorului, CDM, ce vor fi prezentate la § 3.4.4, circuitele de generare a întreruperilor și circuitele de dialog DMA.

FRM cu 8272, CDM și UDF duală formează un SSDF de capacitate dublă față de cel prezentat la § 3.4.1, ajungînd la 1 Moctet. În plus, 8272 oferă posibilitatea lucrului cu 4 UDF, astfel că un SSDF cu două unități duale ajunge la 2 Mocteți pentru un singur controlor, pe dischete cu o singură față.

Vom descrie, în continuare, structura FRM cu 8272, conectat la 4 UDF cu interfață tip UDF 101 (figurile 3.14÷3.16). În afara circuitelor de precompensare, figura 3.15, și bucla PLL, figura 3.16, sînt necesare circuite pentru transferul semnalelor de comandă și stare ale interfeței cu UDF, oscilatorul cu cuarț și generatorul semnalului de sincronizare a scrierii, WCK.

Interfața cu UDF impune decodificarea selectării, $\overline{\text{UNIT SELECT}}$ 0÷3, multiplexarea semnalelor $\overline{\text{UNIT READY}}$ 0÷3, transmiterea, dacă este cazul, a semnalelor $\overline{\text{HEAD LOAD}}$ 0÷3 și rezolvarea decodificării comenzilor/stărilor ce se transmit multiplexat, conform schemei din figura 3.14.

Un circuit demultiplexor dual 1 la 4, tip 74155, distribuie, la UDF aleasă conform codului US0, US1, furnizat de 8272, comanda de „selectat", întotdeauna activă prin fixarea intrării de date 1C la „1" ca și comanda de „încărcare cap", prin aplicarea la intrarea 2C a semnalului HDL, inversat. Întotdeauna, una și numai una dintre cele 4 UDF va fi selectată de unul dintre semnalele $\overline{\text{UNIT SELECT}}$ 0÷3, conform codării US0, US1. Semnalul HDL, amplificat și inversat, se transmite, pentru UDF 101, pe linia $\overline{\text{HEAD LOAD}}$, care ajunge, la toate cele 4 UDF conectate în lanț, *daisy chain*. Schema prevede și conectarea UDF cu unități tip CDC 9401, care au linii separate pentru comanda de încărcare a capului. Pentru această variantă sînt prevăzute semnalele $\overline{\text{HEAD LOAD}}$ 0÷3.

Un multiplexor 4 la 1, tip 74153, alege, conform codului de selecție US0, US1, unul dintre semnalele $\overline{\text{UNIT READY}}$ 0÷3, pentru a fi prezentat pe linia RDY la intrarea 8272. 8272 selectează pe rînd toate cele 4 UDF și monitorizează lansarea întreruperii IT la o schimbare pe oricare dintre liniile $\overline{\text{UNIT READY}}$ 0÷3.

Multiplexarea și demultiplexarea în exteriorul 8272 se aplică și la semnalele de comandă și stare la care se pot separa două faze de funcționare distincte. Faza de deplasare a carului — SEEK — nu interferă în timp cu faza de transfer de date — WR, astfel că se pot separa două seturi de semnale, în funcție de conexiunea externă a 8272, RW/$\overline{\text{SEEK}}$. Comenzile $\overline{\text{STEP}}$, $\overline{\text{DIRECTION}}$ și stările $\overline{\text{TWO SIDED}}$, $\overline{\text{TRACK 00}}$ se aleg în faza SEEK, iar comenzile $\overline{\text{FAULT RESET}}$, $\overline{\text{LOW CURRENT}}$ și stările $\overline{\text{WRITE PROTECT}}$, $\overline{\text{WRITE FAULT}}$ se aleg în faza RW. Multiplexarea/demultiplexarea se implementează cu amplificatoare avînd ieșiri *tri-state*, tip 74125, comandate de linia RW/$\overline{\text{SEEK}}$ (figura 3.14). Comenzile se emit pe interfața cu UDF prin amplifica-

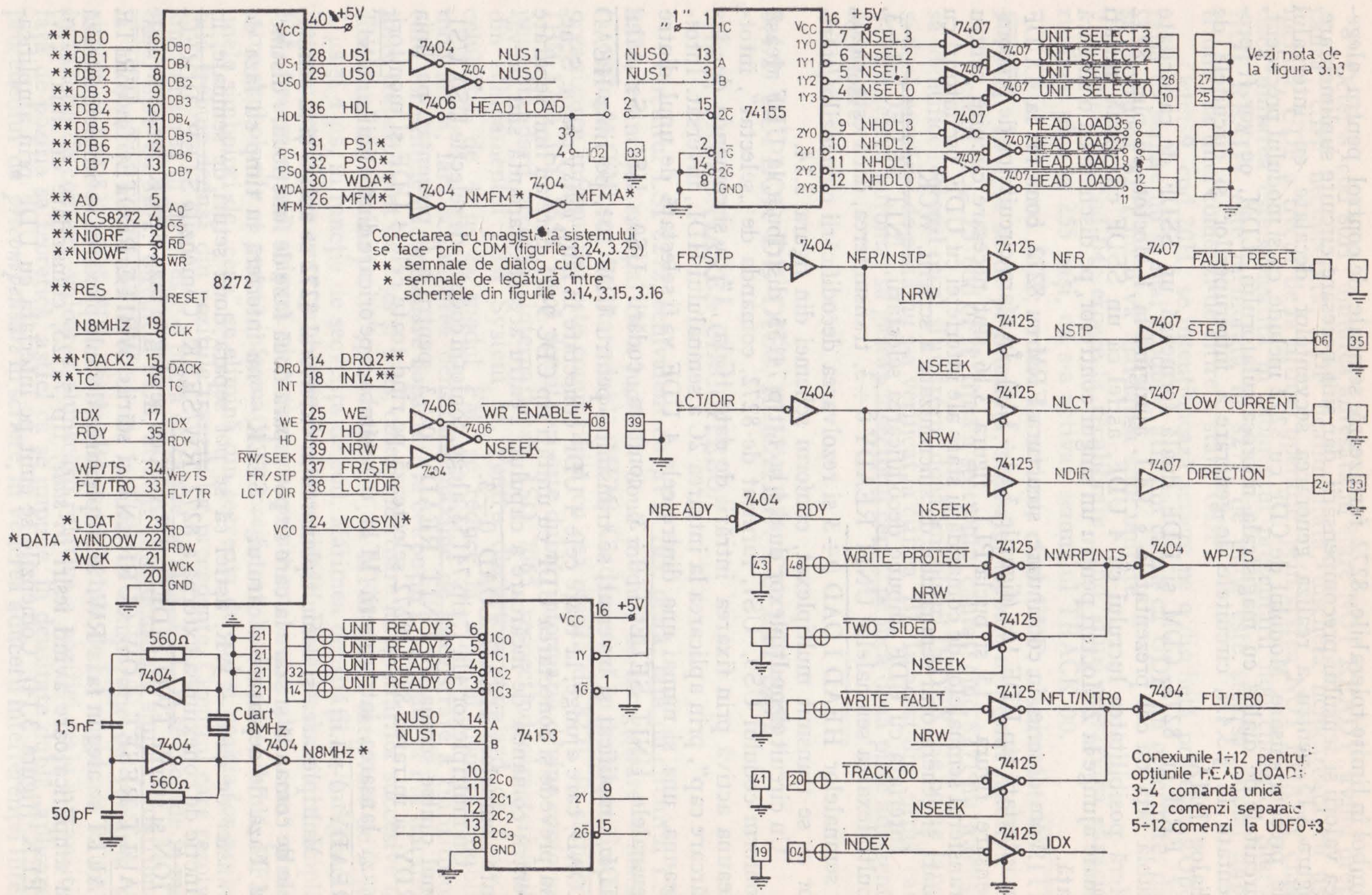


Fig. 3.14. Cuplor de disc flexibil cu 8272, 8272, oscilator, interfața cu UDF(comenzi/stări).

toare cu colector-în-gol neinversoare, pentru ca starea *tri-state* să menţină comanda inactivă la schimbări pe linia $\overline{RW}$/SEEK.

Alte semnale ale interfeţei UDF, care se transmit separat, sînt $\overline{INDEX}$, care ajunge la 8272 ca IDX, WE, care ajunge la interfaţa UDF, ca $\overline{WR\ ENABLE}$ şi HD, util pentru UDF cu 2 capete magnetice, cu posibilitatea de a utiliza ambele feţe ale dischetei (caz în care 4 UDF oferă o memorie externă de 4 Mocteţi).

Pentru dialogul cu circuitul DMA 8257, prezent în CDM, 8272 are prevăzute conexiunile externe DRQ, $\overline{DACK}$ şi TC, prin care se emit cererile de magistrală, se primesc validările pentru fiecare transfer şi, dacă este cazul, notificarea sfîrşitului de secvenţă. Transferurile se fac pe canalul 2 al circuitului DMA.

Transferurile de date, atît pentru scriere, cît şi pentru citire cer rezolvarea unor probleme specifice toleranţelor mai reduse cu care se lucrează în densitate dublă.

La scriere, figura 3.15, se generează semnalul WCK, conform frecvenţei de transfer, se prelucrează semnalul de scriere WDA, şi se transmite pe linia $\overline{WR\ DATA}$ secvenţa cu precompensare prin decalajul poziţiilor de înscriere a tranziţiilor pe suportul magnetic. Un generator cu cuarţ de 8 MHz şi un divizor de frecvenţă realizat cu numărătorul sincron 74193 furnizează semnalele de sincronizare de 8 MHz, 4 MHz, 2 MHz, 1 MHz şi 500 kHz. Pentru densitate

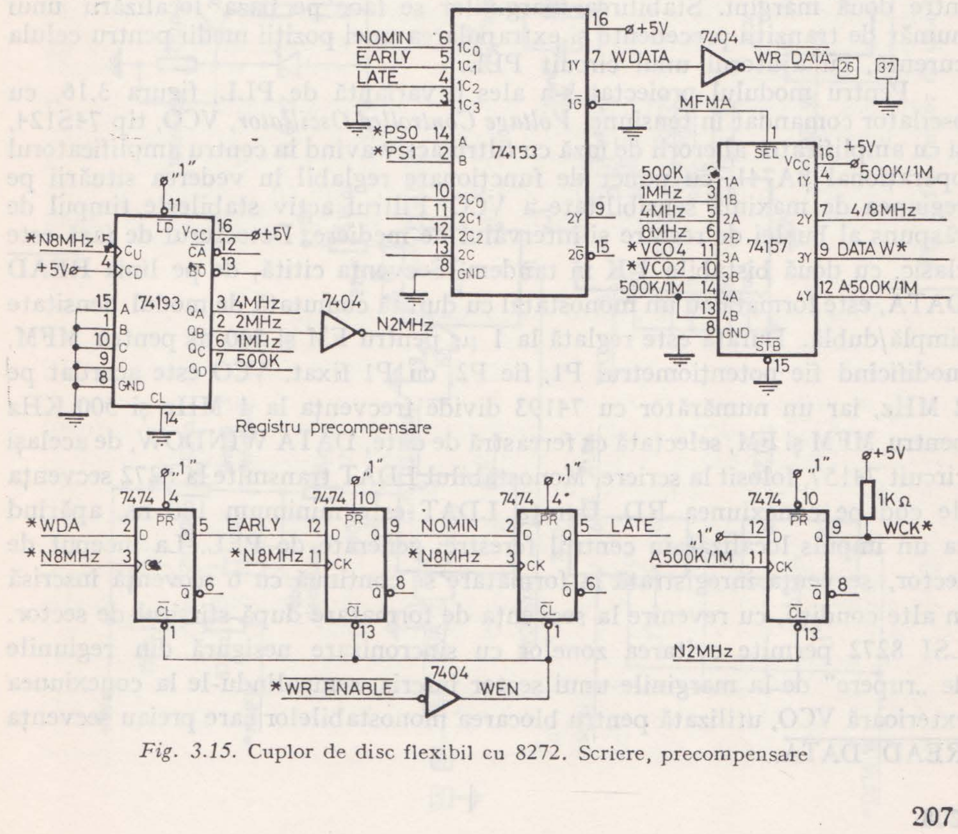


Fig. 3.15. Cuplor de disc flexibil cu 8272. Scriere, precompensare

simplă WCK va primi impulsuri de lățime 250 ns, distanțate la 2 μs. Pentru densitate dublă, același tip de impulsuri vor fi distanțate la 1 μs. Lățimea este dată de perioada semnalului N2MHz, aplicat la resetul unui bistabil 7474, care are drept ceas un semnal cu o frecvență ce stabilește distanța între impulsuri (500 kHz/1 MHz pentru FM/MFM). Circuitul 74157 cu 4 multiplexoare 2 la 1, selectează semnalul de frecvența cerută și întîrzie frontul activ pentru a fi scos din zona în care N2MHz resetează bistabilul 7474.

Conexiunea WDA a 8272 furnizează secvența simbolurilor de cod FM sau MFM, avînd pozițiile tranzițiilor stabilite la distanțe de 2/4 μs în FM și 2/3/4 μs în MFM. Un registru de decalaj realizat cu 7474, figura 3.15, generează secvențele EARLY, NOMIN și LATE. Constanta de precompensare se recomandă 125 ns, pentru pistele exterioare, respectiv 250 ns pentru pistele interioare. Comutarea constantei se poate face cu semnalul LOW CURRENT, după cum acesta este demultiplexat din semnalul LCT/DIR al 8272. Un multiplexor de căi tip 74153, comandat de semnalele de cod ale precompensării PS0, PS1, distribuie simbolurile din secvențele EARLY, NOMIN sau LATE, pentru constituirea secvenței definitive a tranzițiilor ce se vor înscrie pe discul flexibil prin linia de interfață WR DATA. Registrul de decalaj este blocat în absența comenzii de scriere WR ENABLE, cînd se interzic transferuri de impulsuri pe linia WR DATA. Durata impulsurilor este menținută la 250 ns.

La citire, semnalul READ DATA trebuie să ajungă la 8272 însoțit de semnalul DATA WINDOW, care să permită încadrarea impulsurilor de date între două margini. Stabilirea marginilor se face pe baza localizării unui număr de tranziții precedente și extrapolarea unei poziții medii pentru celula curentă, cu ajutorul unui circuit PLL.

Pentru modulul proiectat s-a ales o variantă de PLL, figura 3.16, cu oscilator comandat în tensiune, *Voltage Controlled Oscillator*, VCO, tip 74S124, și cu amplificator al erorii de fază cu filtru activ avînd în centru amplificatorul operațional βA741, cu punct de funcționare reglabil în vederea situării pe regiunea de maximă sensibilitate a VCO. Filtrul activ stabilește timpul de răspuns al buclei de reglare și intervalul de mediere. Detectorul de fază este clasic, cu două bistabile J-K în tandem. Secvența citită, de pe linia READ DATA, este formată cu un monostabil cu durată comutată de modul densitate simplă/dublă. Durata este reglată la 1 μs pentru FM și 500 ns pentru MFM, modificînd fie potențiometrul P1, fie P2, cu P1 fixat. VCO este acordat pe 2 MHz, iar un numărător cu 74193 divide frecvența la 1 MHz și 500 KHz pentru MFM și FM, selectată ca fereastră de date, DATA WINDOW, de același circuit 74157, folosit la scriere. Monostabilul LDAT transmite la 8272 secvența de cod pe conexiunea RD. Durata LDAT este minimum 100 ns, apărînd ca un impuls localizat în centrul ferestrei generate de PLL. La început de sector, secvența înregistrată la formatare se continuă cu o secvență înscrisă în alte condiții, cu revenire la secvența de formatare după sfîrșitul de sector. LSI 8272 permite evitarea zonelor cu sincronizare nesigură din regiunile de „rupere" de la marginile unui sector înscris, semnalîndu-le la conexiunea exterioară VCO, utilizată pentru blocarea monostabilelor care preiau secvența READ DATA.

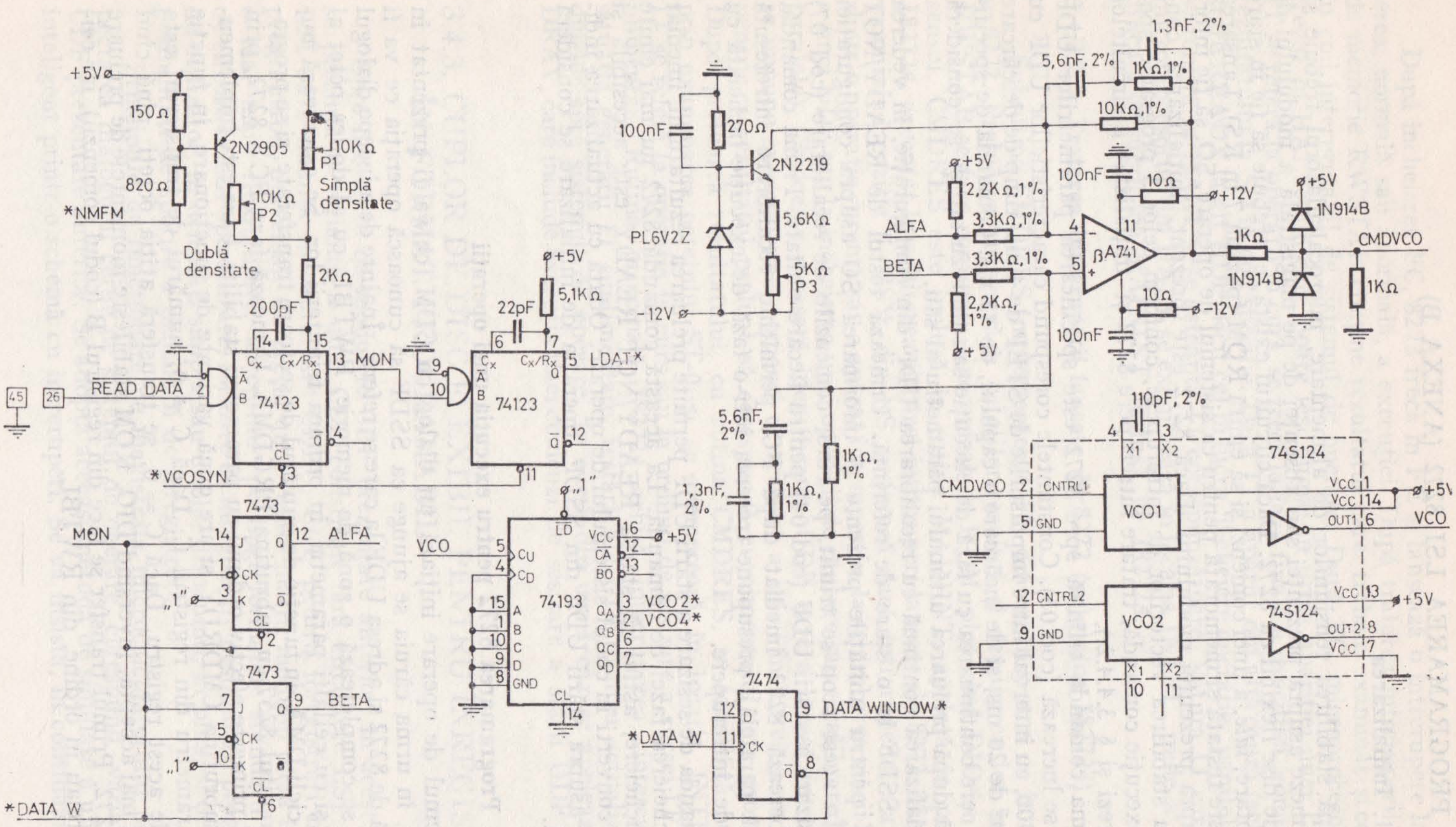


Fig. 3.16. Cuplor de disc flexibil cu 8272. Citire, PLL.

3.4.2.2. PROGRAMAREA LSI 8272 (ANEXA B)

3.4.2.2.1. **Inițializarea**

După stabilirea tensiunilor de alimentare, microcalculatorul trebuie să se informeze asupra prezenței sau absenței de pe magistrală a modulului de cuplor de disc flexibil cu 8272. Dacă circuitul există, el trebuie să fie în stare de așteptare, *idle*, a unei comenzi și să emită RQM (poziția 7 în RS). Lansarea RQM este testată și memorată pentru ca sistemul de operare, SO, să fie informat asupra prezenței modulului. Dacă 8272 este prezent, inițializarea continuă cu stabilirea vectorilor de întrerupere, conform celor 3 programe specifice: execuție comandă, tratare întrerupere 8272 și inițializare a circuitelor CDM (vezi și § 3.4.4.2.).

Prima comandă trimisă spre 8272 este specificarea parametrilor UDF cu care se lucrează (cod 03). Constantele corespund caracteristicilor UDF cu MOM 6400, cu interval între impulsurile de STEP de 4 ms, timp de descărcare a capului de 20 ms, și de încărcare a capului, de 45 ms. Comanda de specificare nu cere continuarea cu fază de execuție sau de rezultat și se consideră încheiată după preluarea ultimului parametru al său.

Inițializarea continuă cu recalibrarea UDF din configurație, în vederea aducerii SSDF la o stare de referință. Urmează testul de READY/NOT READY pentru unitățile prezente și informarea SO asupra configurației SSDF. În acest scop se trimit, pe rînd, comenzile de recalibrare (cod 07) și de sesizare a stării UDF (cod 04) pentru fiecare unitate. Prima comandă, deși eliberează 8272 imediat după FC, permițînd recalibrare întrețesută (comandă paralelă), presupune trecerea într-o fază de execuție încheiată cu lansare de întrerupere.

Comanda de sesizare a stării UDF permite preluarea rezultatului, imediat după încheierea fazei de comandă. La această comandă 8272 nu mai emite IT de încheiere, astfel că starea READY/NOT READY este accesibilă și se poate converti în codul sistemului de operare. Odată cu actualizarea indicatorilor asupra stării UDF din SSDF, operația de inițializare se consideră încheiată.

3.4.2.2.2. **Programarea 8272 pentru execuția unei operații**

Sistemul de operare inițiază un dialog cu CDM, ce va fi prezentat în § 3.4.4.3, în urma căruia se ajunge ca SSDF să cunoască operația ce va fi executată de 8272 și adresa UDF la care se referă. Înainte de a începe dialogul cu 8272, se completează o zonă de memorie, RWTBL, cu al doilea octet al comenzii și cu ceilalți parametri, în ordinea trimiterii lor. Se calculează numărul de cicli DMA, în funcție de numărul de sectoare transferate, și se programează circuitul 8257 cu subrutina PRGDMA. Urmează faza FC a 8272, prin care 8272 primește efectiv ordinul cu parametrii stabiliți. Faza este implementată de subrutina CMDRDY și are două variante de funcționare, în funcție de un parametru din registrul C. Dacă $C \neq 0$, numărul de transferuri este stabilit de acest registru. Dacă $C = 0$, se transferă atîția octeți, pînă cînd 8272 nu-i mai acceptă, inversînd DIO. RQM stabilește momentele de preluare dinspre 8272. Primul transfer se face din registrul B (codul comenzii), iar ceilalți se preiau în ordine, din RWTBL.

După încheierea FC, 8272 trece în FE și lansează o întrerupere la încheierea, normală sau anormală, a execuției. După tratarea întreruperii, o zonă de memorie RWSTAB conține rezultatul care trebuie examinat și convertit în cod interpretabil de sistemul de operare. După FC, microprocesorul este trecut în HALT, din care poate ieși numai în urma apariției unei IT. Dacă IT se referă la operații în curs, se trece la faza finală a execuției, comunicarea rezultatului la SO în cod specific. Dacă IT a apărut din altă cauză (schimbare stare READY la alt UDF decît cel în lucru, încheiere operație „paralelă" etc.), se reintră în HALT și se așteaptă IT corespunzătoare. Natura IT care a scos microprocesorul din HALT se examinează în locația INTFL, unde a fost codificată de subrutina MAININT inclusă în I82724.

3.4.2.2.3. **Tratarea întreruperii lansate de 8272 (subrutina I82724)**

Subrutina CMDRES implementează FR din organigrama de funcționare a 8272. Preluarea succesivă a octeților-rezultat și depunerea lor în tabelul RWSTAB se face continuu, sincronizat de RQM, pînă cînd DIO își schimbă sensul. CMDRES este inclusă în rutina MAININT, apelată în programul I82724, executat la achitarea fiecărei întreruperi lansate de 8272. MAININT, examinînd poziția 7, BSY, din RS, sesizează dacă întreruperea s-a generat dintr-o operație curentă sau a apărut intempestiv, la schimbarea stării READY a unei UDF sau la încheierea unei operații de căutare sau recalibrare (comenzi paralele). Cînd este necesară identificarea UDF care a lansat întreruperea, se impune execuția unei comenzi de sesizare a stării în întrerupere (cod 08) pentru ca rezultatul preluat să corespundă evenimentului ce a provocat IT. După preluarea rezultatului, cu rutina CMDRES, se identifică sursa IT, prin examinarea octetului ST-0 și se memorează în locația INTFL dacă IT se referă la operația în curs sau la ordinul de sesizare a sursei IT date suplimentar. Rutina principală I82724 achită întreruperea, intercalînd, în situația generării IT suplimentare, testări ale UDF pentru actualizarea codurilor spre sistemul de operare. În acest din urmă caz se apelează rutina DRAV, care include execuția unei comenzi de sesizare a stării UDF (cod 04).

3.4.3. CUPLOR DE DISC FLEXIBIL PENTRU ÎNREGISTRĂRI ÎN FORMAT PROGRAMABIL

Spre deosebire de modulele CDF prezentate anterior, în acest paragraf se prezintă o variantă a unui CDF constituit fără circuitele LSI 8271 sau LSI 8272 numită în continuare CDFU, CDF pentru înregistrări în format programabil, universal. Cu prețul unui volum mărit, modulul oferă în schimb o mai mare flexibilitate, acceptînd, pe lîngă formatele standard asigurate de circuitele LSI specializate, formate de înregistrare nestandard programabile de utilizator, în densități simplă sau dublă, dispunînd, în plus, și de anumite posibilități de sporire a densității de înregistrare.

Principiile de funcționare pe magistrala microcalculatorului pentru respectarea dialogului cerut de sistemul de operare, corespund variantelor de CDF prezentate anterior. Se adoptă aceeași metodă de simulare a unui cuplor inteligent printr-o schemă cu întreruperi. Se va insista, în continuare, numai

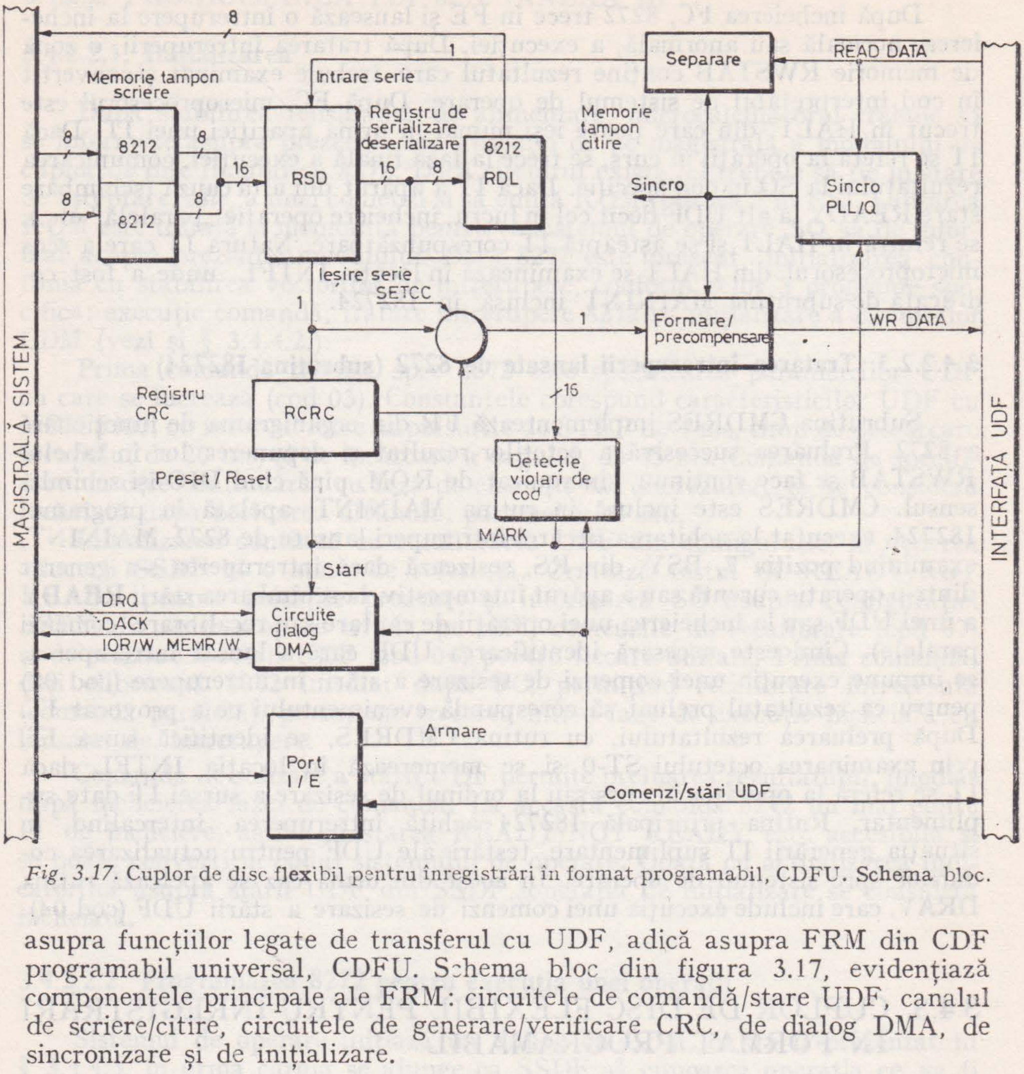


Fig. 3.17. Cuplor de disc flexibil pentru înregistrări în format programabil, CDFU. Schema bloc.

asupra funcțiilor legate de transferul cu UDF, adică asupra FRM din CDF programabil universal, CDFU. Schema bloc din figura 3.17, evidențiază componentele principale ale FRM: circuitele de comandă/stare UDF, canalul de scriere/citire, circuitele de generare/verificare CRC, de dialog DMA, de sincronizare și de inițializare.

3.4.3.1 CIRCUITELE DE COMENZI/STĂRI

Cele mai multe dintre semnalele interfeței cu UDF sînt direct accesibile printr-un *port* 8255, pentru realizarea condițiilor necesare transferului: selectarea, accesul la pistă, „încărcarea“ capului, lansarea comenzilor specifice scrierii (dacă este cazul), sesizarea diferitelor condiții de lucru, ca în schema din figura 3.18. Toate aceste semnale nu au condiții critice de timp și pot fi manevrate prin instrucțiuni de I/E ale microprocesorului, cu durate de execuție de ordinul μs.

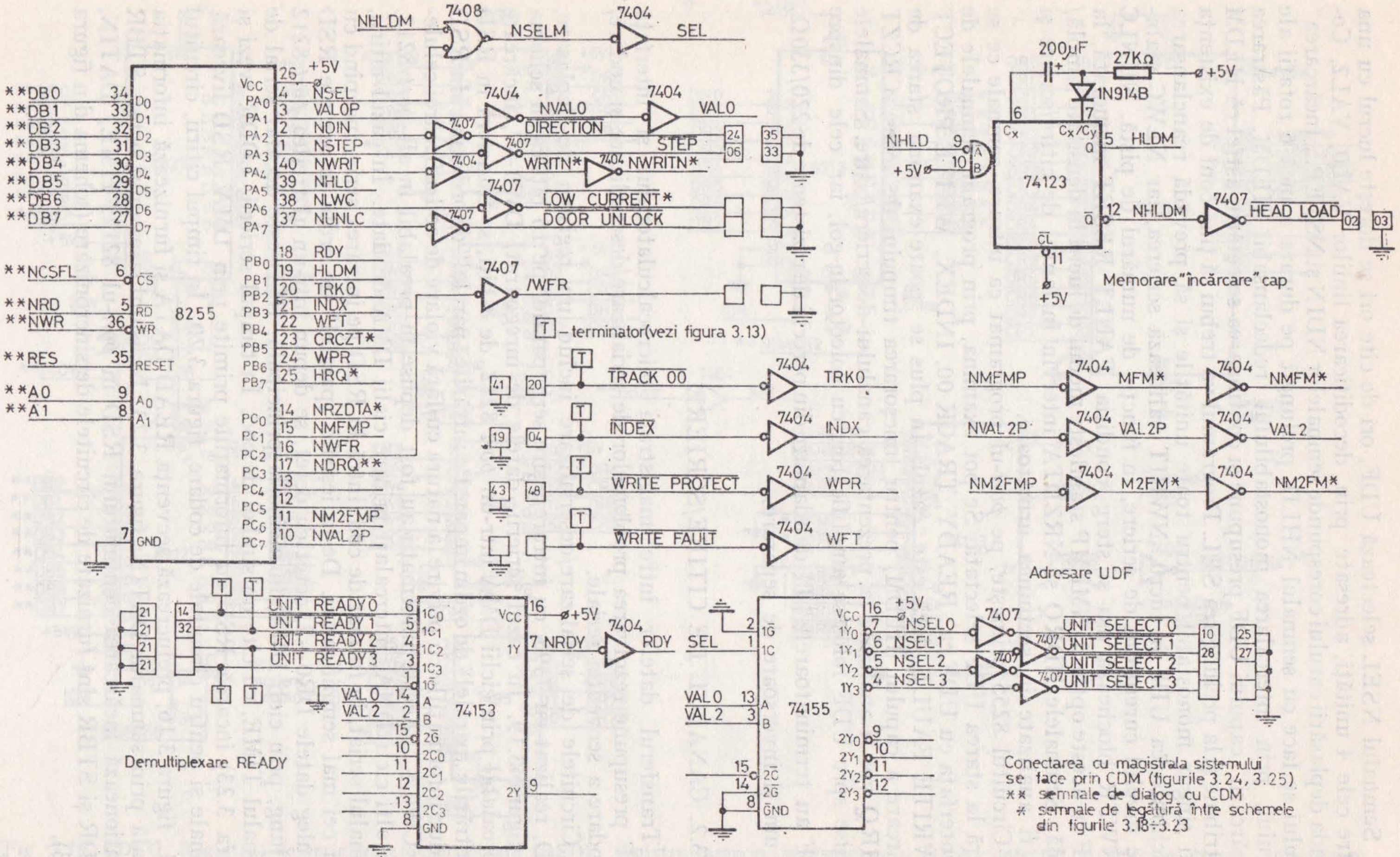


Fig. 3.18. CDFU. Interfaţa cu UDF (comenzi/stări).

Semnalul NSEL selectează UDF, ori de cîte ori se dorește lucrul cu una dintre cele 4 unități, adresate prin decodificarea liniilor VAL0, VAL2. Comanda deplasării carului corespunde semnalelor NDIN și NSTEP. „Încărcarea" capului se face cu semnalul NHLD, prelungit pe durata a încă 6 rotații ale discului, prin intercalarea monostabilului redeclanșabil HLDM. Păstrarea „încărcării" capului UDF presupune și continuarea selectării, astfel că HLDM contribuie și la prelungirea SEL. Programul va trebui să țină cont de existența unui singur monostabil pentru toate unitățile și să prevadă reanclanșarea la schimbarea UDF în lucru. NWRIT validează scrierea, iar NLWC stabilește nivelul curentului de scriere, în funcție de numărul de pistă. NUNLC și NWFR blochează ușa și șterg condiția FAULT RESET memorată, la UDF cu aceste opțiuni. NMFMP stabilește modul de lucru în densitate simplă/dublă. Semnalele NDRQ și NRZDTA intervin în lanțul de citire/scriere și vor fi analizate în secțiunea următoare.

Circuitul 8255 primește, pe *port*-ul programat ca intrare, semnale ce se referă la starea UDF selectată. Se pot examina, prin program, semnalele de pe interfața cu UDF ca: READY, TRACK 00, INDEX, WRITE PROTECT și WRITE FAULT, dacă este cazul. În plus se poate examina starea de încărcare a capului, HLDM, pentru micșorarea timpului de acces. CRCZT și HRQ vor fi analizate la prezentarea canalului de scriere/citire. Semnalele trimise spre UDF folosesc amplificatori cu colector-în-gol, iar cele dinspre UDF au terminatoarele $|\overline{T}|$ de adaptare din perechi de rezistențe de 220/330Ω, precum și inversoare de separare.

3.4.3.2. CANALUL DE CITIRE/SCRIERE

Transferul datelor între magistrala microcalculatorului și interfața UDF presupune rezolvarea problemelor de serializare/deserializare și codare/decodare a secvenței seriale.

Circuitele de serializare/deserializare includ un registru de deplasare, RSD, realizat cu 7495, cu intrare/ieșire serie/paralel pe 16 biți ca în schema din figura 3.19. În cazul scrierii, datele de înregistrat, $DW0 \div 7$, încărcate în prealabil prin ciclii DMA într-un *port* 8212, de 8 biți, se transferă în RSD pe intrările „paralel" de ordin impar. Pe intrările „paralel" de ordin par ale RSD se aduc informațiile cu privire la natura cu/fără violare de cod a fiecărei celule-bit, $CW0 \div 7$. Aceste informații au fost depuse în prealabil în alt *port* 8212, prin alți cicli DMA, intercalați printre ciclii DMA de date. În cazul citirii, semnalul serial, în format de cod, intră în RSD pe intrarea serie, începînd cu biții cei mai semnificativi. De la ieșirile „paralel" de ordin impar ale RSD se culeg datele $DR0 \div 7$, octet cu octet, se depun într-un al treilea *port* 8212 și ajung, prin cicli DMA, în memoria principală. RSD este decalat serial de semnalul TIME, cu cîte un simbol de cod. În timpul scrierii, LODR (vezi și figura 3.23) încarcă RSD cu informațiile primite prin DMA. RSD livrează semnale și pentru circuitele de codare, figura 3.20. În timpul citirii, circuitul PLL, figura 3.16, prelucrează secvența $\overline{\text{READ DATA}}$ și furnizează informația serială prin semnalul DATIN, figura 3.23. La asamblarea unui octet, STBR eșantionează încărcarea conținutului RSD în *port*-ul 8212. TIME, DATIN, LODR și STBR sînt furnizate de circuitele de sincronizare (schema din figura 3.23).

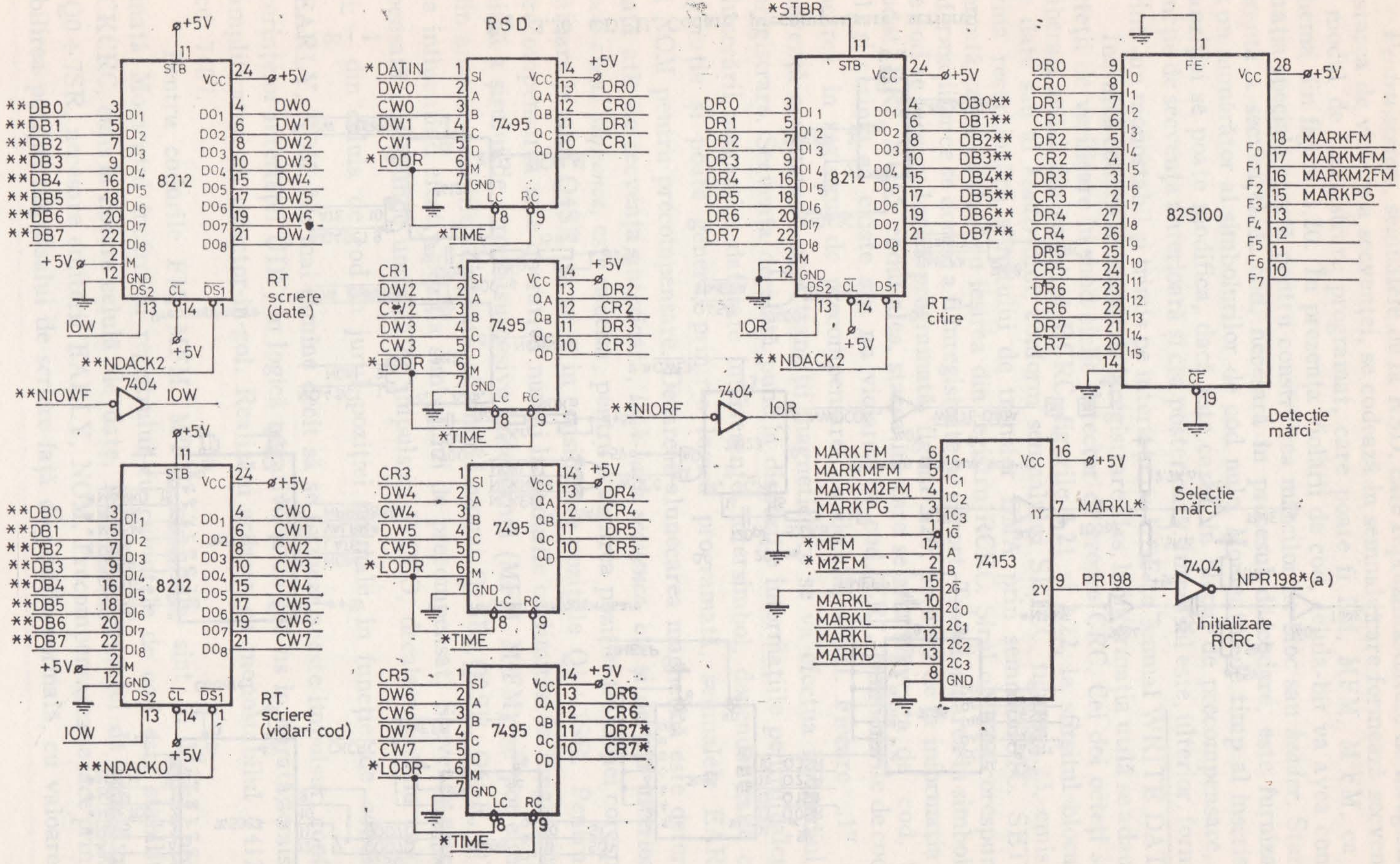


Fig 3.19. CDFU. Registrul de serializare/deserializare (RSD), registre tampon accesate prin DMA, detecție și selecție marcă, inițializare registru CRC (RCRC)

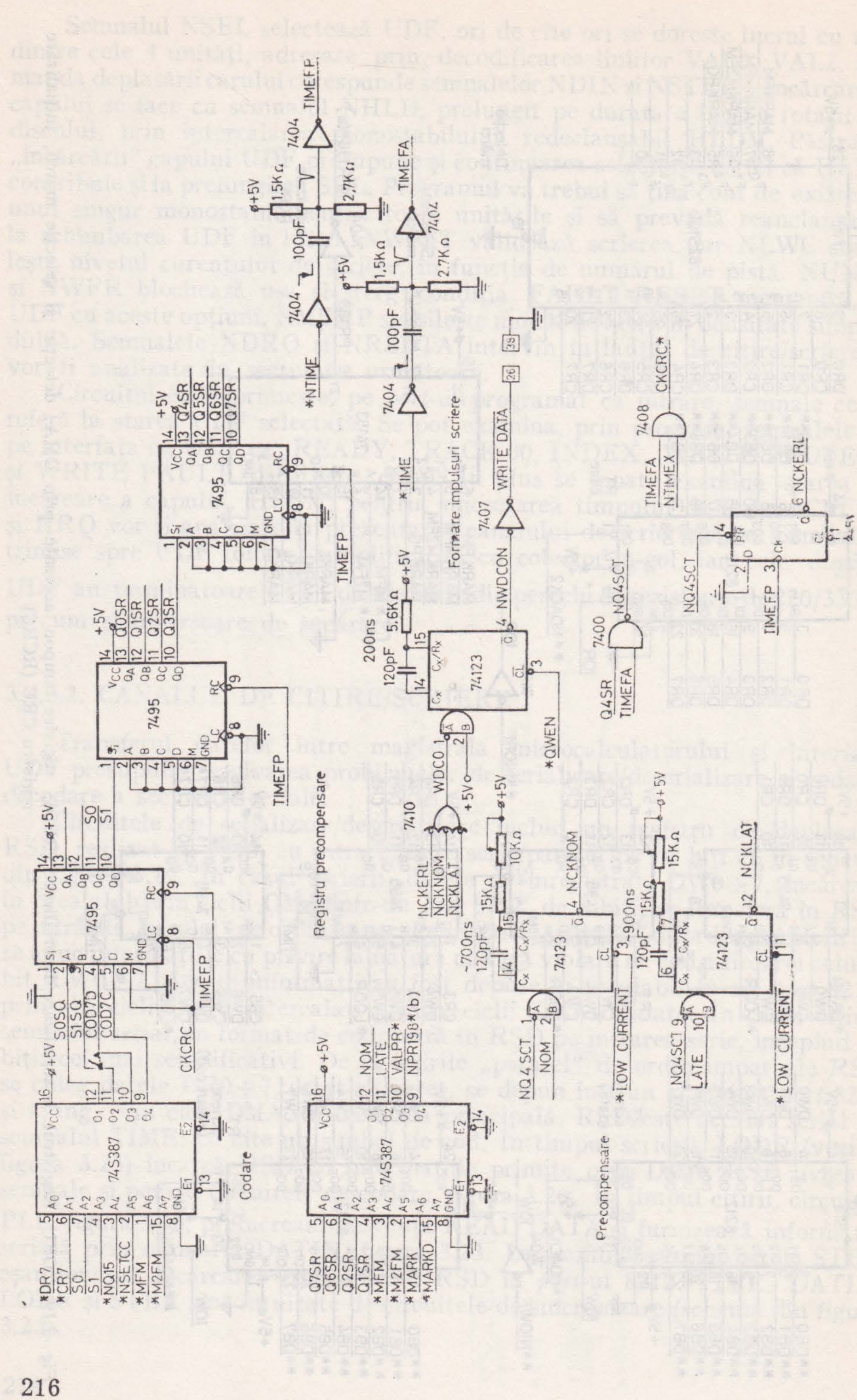


Fig. 3.20. CDFU. Codare, precompensare, scriere

Pentru scriere, semnalele de la RSD, care reprezintă datele de înregistrat și starea de violare a secvenței, se codează în semnale care formează secvența în modul de înregistrare programat, care poate fi FM, MFM, M^2FM, ca în schema din figura 3.20. În prezența violării de cod, celula-bit va avea configurația specială cerută pentru construirea mărcilor de bloc sau *header*. Starea curentă a secvenței de cod, necesară în procesul de codare, este furnizată de un numărător al simbolurilor de cod nule. Momentul de timp al înscrierii tranziției se poate modifica, dacă este cazul, în blocul de precompensare, în funcție de secvența anterioară și cea posterioară. Impulsul este, ulterior, format printr-un monostabil și trimis pe interfața cu UDF ca semnal WRITE DATA.

Înaintea codării în modul de înregistrare ales, la informația utilă se adaugă octeții de verificare în mod ciclic detector de eroare CRC. Cei doi octeți sînt eliberați serial din registrul RCRC, figurile 3.21 și 3.22, la sfîrșitul blocului de date sau al *header*-ului, conform semnalului SETCC, figura 3.23, emis în urma recepționării sfîrșitului de transfer DMA prin semnalul TC. SETCC comută calea de date cu ieșirea din registru RCRC. Șirul obținut corespunde informațiilor ce se doresc a fi înregistrate pe suport. Transformarea în simboluri de cod se face cu logică programată, figura 3.20, pornind de la informația de codat, DR7, NQ15, codul ales, starea în care se află secvența de cod, S0, S1, și cerința să existe sau nu violarea de cod, CR7. Simbolurile de cod se încarcă în registrul de precompensare, cu decalaj serial. Fiecare „1" din secvență corespunde unei tranziții magnetice ce se va efectua în mediul de înregistrare. Secvența decalată pune la dispoziție informațiile pentru calculul alunecării magnetice datorate interferenței intersimbol, dependente de configurație și poate genera, printr-o logică programată, semnalele EARLY și NOM pentru precompensare. Deoarece alunecarea magnetică este determinată atît de secvența anterioară, *look-back sequence*, cît și de cea ulterioară, *look-ahead sequence*, este necesar, pentru stabilirea poziției tranziției corespunzătoare bitului Q4SR, să se ia în considerare pozițiile Q1 ÷ 7SR. Pentru că precompensarea este operativă numai la codurile cu limitări la lungimea minimă a șirului de zerouri succesive în secvență (MFM, M^2FM), se pot elimina din analiză pozițiile adiacente, rămînînd numai Q1SR, Q2SR, Q6SR, Q7SR ca influențînd efectiv poziția simbolului de precompensat. Secvența precompensată corespunde unui șir de impulsuri WDCO, decalate sau nu cu circa $\pm\frac{1}{8}$ din celula de cod în jurul poziției normale, în funcție de condițiile EARLY, NOM. Nu mai rămîne decît să se formeze aceste impulsuri conform cerințelor interfeței UDF, în logică negativă și de 200 ns în durată, emise de amplificatori cu colector-în-gol. Realizăm aceasta cu monostabilul 74123 și cu 7407.

Pentru codurile FM, MFM, M^2FM, circuitele sînt în logică programată. Momentul încărcării registrului cu cuvintele de cod este stabilit de CKCRC, emis la fiecare celulă de date. Registrul secvenței de cod, livrează Q0 ÷ 7SR, necesare emiterii EARLY, NOM. Precompensarea se face prin stabilirea poziției impulsului de scriere față de poziția normală, cu valoare sta-

bilitâ din constantele RC ale celor două monostabile, CKNOM şi CKLAT. Se consideră valoarea de referinţă dată de frontul căzător al CKNOM faţă de frontul urcător al semnalului TIME. Q4SR şi TIMEFP formează cu un bistabil semnalul CKERL pentru înregistrarea în avans. Pentru corecţia cu întîrziere, la apariţia condiţiei EARLY, CKLAT are o întîrziere de circa 900 ns faţă de frontul urcător al semnalului TIME. Se obţine WDCO, avînd front coborîtor la 500, 700, 900 ns faţă de referinţa TIME. Toată secvenţa de cod se consideră întîrziată cu 700 ns. De menţionat că diferenţa de 200 ns poate fi reglată separat pentru corecţia în avans şi pentru cea cu întîrziere, în funcţie de caracteristicile UDF. Pentru UDF 101 cu CDC 9404, fabricantul recomandă constante de precompensare apropiate de 250 ns.

PROM-ul de precompensare şi control ciclic (figura 3.20) este împărţit în 16 zone, în funcţie de semnalele MARKD, MARK, M2FM, MFM:

	Intrări								Ieşirii			
	7	6	5	4	3	2	1	0	3	2	1	0
Adrese	MARKD	MARK	M2FM	MFM	Q1SR	Q2SR	Q6SR	QSR	NPR198	VALPR	LATE	NOM
0 ÷ 15	0	0	0	0	×	×	×	×	0	1	0	1
16 ÷ 31	0	0	0	1					0	1		
32 ÷ 47	0	0	1	0					0	0		
48 ÷ 63	0	0	1	1					0	1		
64 ÷ 79	0	1	0	0	×	×	×	×	1	1	0	1
80 ÷ 95	0	1	0	1					1	1		
96 ÷ 111	0	1	1	0					1	0		
112 ÷ 127	0	1	1	1					0	1		
128 ÷ 143	1	0	0	0	×	×	×	×	0	1	0	1
144 ÷ 159	1	0	0	1					0	1		
160 ÷ 175	1	0	1	0					0	0		
176 ÷ 191	1	0	1	1					1	1		
192 ÷ 207	1	1	0		×	×	×	×	1	1	0	1
208 ÷ 223	1	1	0	1					1	1		
224 ÷ 239	1	1	1	0					1	0		
240 ÷ 255	1	1	1	1					1	1		

În densitate simplă, $\overline{\mathrm{MFM}} \cdot \overline{\mathrm{M2FM}} = 1$, nu se aplică precompensarea. Pentru celelalte trei moduri se aplică precompensarea, la fel în fiecare caz. Pentru cele 16 combinații posibile ale semnalelor Q x SR, x = 1, 2, 6, 7, se obține tabelul de mai jos:

3	2	1	0	1	0
Q1SR	Q2SR	Q6SR	Q7SR	LATE	NOM
0	0	0	0	0	1
0	0	0	1	0	0*
0	0	1	0	0	0*
0	0	1	1	0	1
0	1	0	0	1	0
0	1	0	1	1	0
0	1	1	0	0	1
0	1	1	1	0	1
1	0	0	0	1	0
1	0	0	1	0	1
1	0	1	0	0	0*
1	0	1	1	0	1
1	1	0	0	0	1
1	1	0	1	0	1
1	1	1	0	0	1
1	1	1	1	0	1

Notă: * semnifică o combinație EARLY.

Ieșirile pentru RCRC, VALPR și NPR 198, corespund ecuațiilor:

$$\mathrm{VALPER} = \mathrm{MFM} + \overline{\mathrm{M2FM}}$$

$$\mathrm{PR198} = \mathrm{MARK}\ (\overline{\mathrm{MFM}} + \overline{\mathrm{M2FM}}) + \mathrm{MARKD} \cdot \mathrm{MFM} \cdot \mathrm{M2FM} = \\ = \overline{\mathrm{NPR\,198}}$$

PROM-ul de codare se explicitează prin ecuațiile logice ale semnalelor S1SQ, S0SQ, COD7D, COD7C. Pentru toate modurile de codare, automatul secvenței funcționează ca numărător al șirului de zerouri consecutive, fiind incrementat la fiecare simbol „0“ și inițializat la fiecare simbol „1“:

$$\mathrm{S0SQ} = \overline{\mathrm{DR7}} \cdot \mathrm{CR7} + \overline{\mathrm{DR7}} \cdot \mathrm{S0}$$

$$\mathrm{S1SQ} = \overline{\mathrm{DR7}} \cdot \overline{\mathrm{CR7}} \cdot \mathrm{S1}$$

Pentru fiecare mod de inregistrare, codarea are aceeași ecuație a simbolului de „date“, COD7R = DR7, și cîte o ecuație specifică pentru „ceasuri“:

$$\mathrm{FM}: \quad \mathrm{COD7C} = \mathrm{CR7}$$

$$\mathrm{MFM}: \quad \mathrm{COD7C} = \overline{\mathrm{DR7}} \cdot \mathrm{CR7}\,(\mathrm{S0} + \mathrm{S1})$$

$$\mathrm{M2FM}: \quad \mathrm{COD7C} = \overline{\mathrm{CR7}} + \overline{\mathrm{DR7}} \cdot \mathrm{CR7} \cdot \mathrm{S1}$$

Ecuația reunită pentru toate modurile devine:

$$\mathrm{COD7C} = \mathrm{CR7} \cdot \overline{\mathrm{MFM}} \cdot \overline{\mathrm{M2FM}} + \overline{\mathrm{DR7}} \cdot \mathrm{CR7} \cdot \mathrm{S0} \cdot \mathrm{MFM} \cdot \overline{\mathrm{M2FM}} + \\ + \overline{\mathrm{DR7}} \cdot \mathrm{CR7} \cdot \mathrm{S1} \cdot \mathrm{MFM} + \overline{\mathrm{DR7}} \cdot \mathrm{CR7} \cdot \mathrm{S1} \cdot \mathrm{M2FM} + \overline{\mathrm{CR7}} \cdot \mathrm{M2FM}$$

Ecuațiile finale pentru COD7D și COD7C se obțin înlocuind DR7 prin:

$$\mathrm{DR7 \cdot NSETCC} + \overline{\mathrm{NQ15}} \cdot \overline{\mathrm{NSETCC}}$$

în scopul includerii octeților CRC.

Semnalul CR7 evidențiază violările de cod. Acestea apar la CR7 = 0 și se manifestă prin lipsă de impuls la FM, MFM și adăugare de impuls la M^2FM.

Schema de codare se poate îmbunătăți pentru implementarea și a altor coduri RLL, mai performante, ca, de exemplu, codurile (2,7) 3PM sau codurile ternare ([15] cap. 2), cu păstrarea principiului de funcționare, prin adaptarea la lungimile și simbolurile de cod cerute și schimbarea programării PROM-urilor. Se poate obține astfel o mărire a densității de înregistrare, de 3 sau 4 ori mai mare decît densitatea simplă (FM), cu păstrarea densității de tranziții în magnetizare. Evident, canalul de citire va fi adaptat algoritmului de decodificare, ceva mai complicat, cu tabelă de conversie memorată tot în PROM.

3.4.3.3. CIRCUITUL DE CONTROL CU COD CICLIC DETECTOR DE EROARE

Pentru a se evita lansarea unor operații cu unitatea centrală avînd la bază programe sau date incorecte, ceea ce ar periclita funcționarea în condiții normale a întregului microcalculator, trebuie să se prevadă semnalarea eventualelor erori la citirea de pe discul flexibil. În cazul detectării unei astfel de erori, sistemul de operare încearcă o nouă citire pe același sector, după ce a reinițializat SSDF printr-o operațiune de recalibrare. Abia după ce se constată că eroarea persistă după un număr stabilit de reluări, se decide abandonarea comenzii, cu semnalarea defectului la consola operatorului. Pentru ca SSDF să poată detecta erorile de citire se completează cîmpul de date cu simboluri redundante, folosind cod ciclic, *Cyclic Redundancy Check*, CRC. Funcționarea în mod generator/verificator se poate face cu un registru de deplasare, RCRC, figurile 3.21, 3.22, *modulo* un polinom ireductibil. Pentru discurile flexibile standard se folosește polinomul $p(x) = x^{16} + x^{12} + x^5 + 1$. Registrul ca generator, adaugă 2 octeți, în urma blocului, adică un polinom de grad 15, astfel încît polinomul echivalent total să devină multiplu de $p(x)$. Fiecare termen al polinomului are exponentul conform poziției relative în bloc și coeficientul conform conținutului locației. Ponderile exponenților descresc de la începutul de bloc spre sfîrșit, în ordinea de scriere a octeților iar în cadrul fiecărui octet, descresc conform ponderilor de bit. Polinomul unui sector, care cuprinde octetul de marcă, 128 octeți de date și 2 octeți CRC, va avea gradul 1 047. La verificare se controlează dacă la citirea întregului bloc, inclusiv a octeților de control, se obține anularea registrului *modulo* $p(x)$, ceea ce denotă divizibilitatea cu polinomul generator. Dacă registrul nu se anulează, înseamnă că s-a adăugat, în mod intempestiv, la polinomul original al datelor, un polinom de eroare, în general nedivizibil cu polinomul de control. Eroarea este detectată și memorată, apariția ei fiind testată la încheierea oricărei operații de citire de pe discul flexibil.

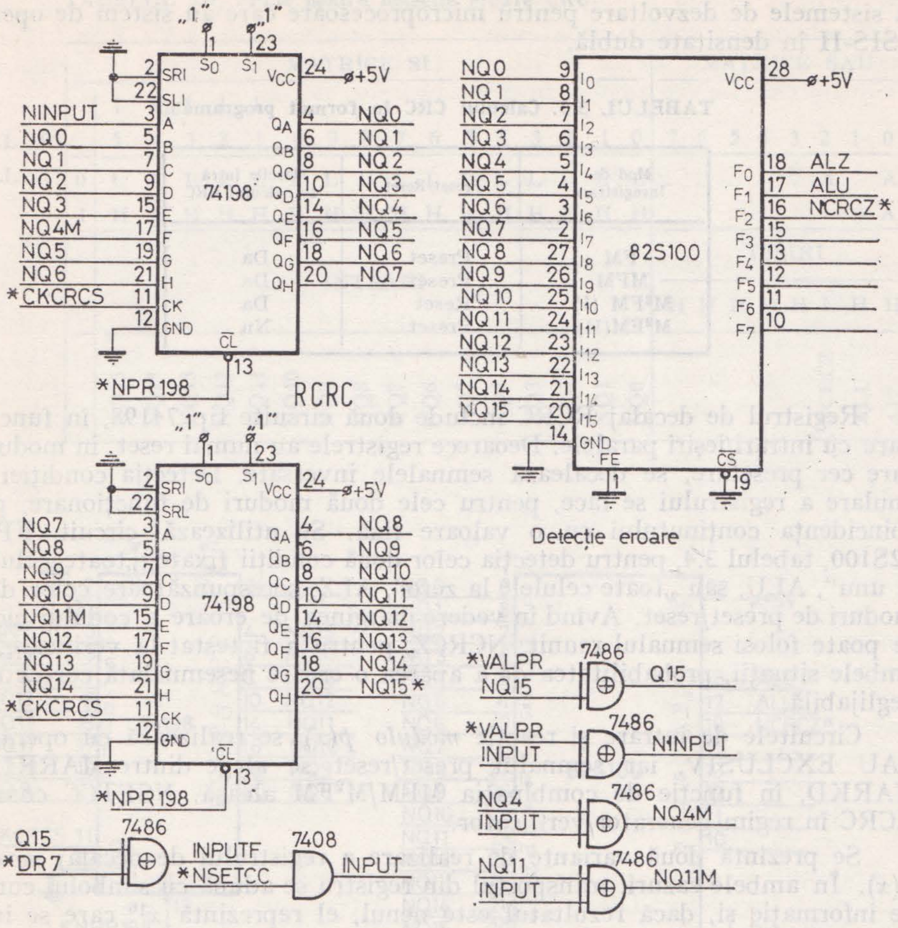


Fig. 3.21. CDFU. Registrul CRC (RCRC). Varianta 1.

Pentru a se preîntîmpina funcționarea banală, în cazul blocurilor de date cu conținut nul, se folosește preînmulțirea polinomului, ce intră în calcul, cu o valoare fixă corespunzătoare pornirii cu registrul *modulo* $p(x)$ presetat. În plus, în anumite cazuri, se pot lua în calcul octeții de marcă, atît la sector cît și la BI. Registrul RCRC este proiectat să funcționeze pentru mai multe formate de înregistrare, în densitate simplă sau dublă. În densitate simplă, format standard ISO 5654, RCRC este presetat la sosirea mărcii. Calculul începe cu octetul de marcă, se continuă pe blocul de date propriu-zis și se încheie cu octeții CRC. În densitate dublă, mod MFM, format standard ISO 7065, intră în calcul toți cei 4 octeți de marcă. Pentru modul de lucru în densitate dublă, mod M^2FM, sînt prevăzute două variante de funcționare: mod H, cu presetare, dar fără considerarea octetului unic de marcă și mod I, cu resetarea registrului la sosirea octetului unic de marcă, acesta intrînd în calculul CRC. Modul H corespunde SSDF de tipul 9885M al firmei Hewlett-Packard, iar modul I pentru SSDF ale firmei INTEL,

la sistemele de dezvoltare pentru microprocesoare care au sistem de operare ISIS-II în densitate dublă.

TABELUL 3.3. **Calculul CRC în format programabil**

Mod de înregistrare	Preset/Reset	Mărcile intră în calculul CRC
FM	Preset	Da
MFM	Preset	Da
M^2FM (I)	Reset	Da
M^2FM(H)	Preset	Nu

Registrul de decalaj RCRC include două circuite tip 74198, în funcționare cu intrări/ieșiri paralele. Deoarece registrele au numai reset, în modurile care cer presetare, se decalează semnalele inversate. Detecția condiției de anulare a registrului se face, pentru cele două moduri de funcționare, prin coincidența conținutului cu o valoare fixă. Se utilizează circuit FPLA 82S100, tabelul 3.4, pentru detecția celor două condiții fixate, „toate celulele la unu", ALU, sau „toate celulele la zero", ALZ, corespunzătoare celor două moduri de preset/reset. Avînd în vedere marginea de eroare a codului ciclic, se poate folosi semnalul reunit, NCRCZ, pentru a fi testat la verificare. În ambele situații, probabilitatea de a apărea o eroare nesemnalată corect este neglijabilă.

Circuitele de intrare și reacție *modulo* $p(x)$ se realizează cu operatori SAU EXCLUSIV, iar semnalul preset/reset se alege dintre MARKL și MARKD, în funcție de combinația MFM/M^2FM aleasă. NSETCC comută RCRC în regim generator/verificator.

Se prezintă două variante de realizare a registrului de decalaj *modulo* $p(x)$. În ambele cazuri, transportul din registru se adună cu simbolul curent de informație și, dacă rezultatul este nenul, el reprezintă x^{16} care se înlocuiește cu $x^{12} + x^5 + 1$. În prima variantă, figura 3.21, înlocuirea se face într-un singur tact, la conținutul registrului, decalat cu o poziție, adunîndu-se polinomul $x^{12} + x^5 + 1$ și înscriindu-se rezultatul înapoi în registru. Operația se reia la fiecare simbol de informație care intră în calcul. În lectură, se citesc și cei 2 octeți CRC, după care conținutul nenul al registrului semnalează eroare. În scriere, cînd cei 2 octeți CRC trebuie să fie generați, după trecerea blocului de date și calcularea restului, ca mai sus, se blochează mecanismul de reacție cu $x^{12} + x^5 + 1$ și se decalează conținutul, spre a fi adăugat în urma blocului. Regimul de generator apare în urma semnalării TC la sfîrșit de transferuri DMA, care declanșează bistabilul SETCC, figura 3.23, sincronizat pe semnalul de încheiere a asamblării unui octet, B7TIM, dacă NRZDTA, permite, prin program, acest lucru. După înscrierea celor 2 octeți CRC, registrul revine la regimul de verificare. În a doua variantă, figura 3.22, înlocuirea x^{16} prin $x^{12} + x^5 + 1$ se face în 16 tacte succesive. În urma recepției ultimului bit de informație, registrul de deplasare RCRC conține restul într-o formă modificată. La citire, ca și în varianta precedentă, după recepționarea ultimului simbol din bloc, registrul de deplasare va fi nul, în condițiile unui transfer fără eroare. În regim de generator,

TABELUL 3.4. **FPLA pentru detecție eroare CRC**

		MATRICE SI																MATRICE SAU							
ALL „0“		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
ALL „1“	0	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	.	.	.	.	.	A	.	A
	1	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	.	.	.	.	.	.	A	A
		INTRĂRI																IEȘIRI							
																		H	H	H	H	H	L	H	H
		NQ15	NQ14	NQ13	NQ12	NQ11	NQ10	NQ9	NQ8	NQ7	NQ6	NQ5	NQ4	NQ3	NQ2	NQ1	NQ0						NCRCZ	ALU	AIZ

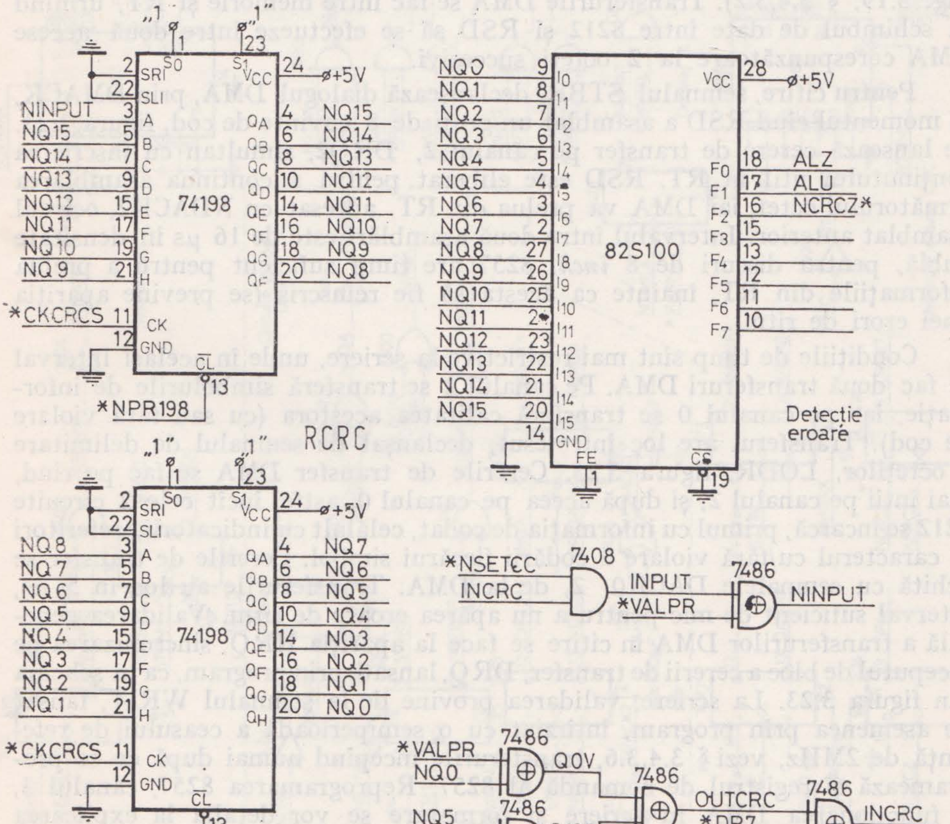


Fig. 3.22. CDFU. Registrul CRC (RCRC). Varianta 2

după ultimul simbol de informație, la apariția semnalului SETCC, se blochează schema de reacție și se emit, la ieșirea circuitului SAU EXCLUSIV al pozițiilor 12, 5, 0, de la RCRC, în 16 tacte succesive, cele 16 simboluri de control care se vor adăuga în urma blocului de date.

3.4.3.4. CIRCUITELE DE DIALOG DMA

Circuitul 8257 dirijează transferurile cu memoria, interacționînd cu registrul de serializare/deserializare, RSD, pe partea de încărcare/preluare în paralel a informațiilor. Funcționarea CDF presupune folosirea a 3 canale DMA: canalul 2 dirijează fluxul de date, atît în scriere, cît și în citire, canalul 3 este folosit ca registru temporar pentru funcționarea canalului 2 în regim *autoload*, iar canalul 0 trimite, numai la scriere, indicațiile cu privire la caracterul cu/fără violare al simbolurilor transferate pe canalul 2. La scriere, canalele 0 și 2 funcționează înlănțuit, cu prioritate rotativă.

Trei registre tampon RT, tip 8212, de cîte un octet, se folosesc la interfața între magistrala de date și registrul de serializare/deserializare RSD. (fig. 3.19, § 3.4.3.2). Transferurile DMA se fac între memorie și RT, urmînd ca schimbul de date între 8212 și RSD să se efectueze între două accese DMA corespunzătoare la 2 octeți succesivi.

Pentru citire, semnalul STBR declanșează dialogul DMA, prin DMACK, în momentul cînd RSD a asamblat un grup de 8 cuvinte de cod, figura 3.23. Se lansează cerere de transfer pe canalul 2, DRQ2, simultan cu înscrierea conținutului util în RT. RSD este eliberat pentru a continua asamblarea următorului octet, iar DMA va prelua din RT, adresat cu NDACK2, octetul asamblat anterior. Intervalul între două asamblări este de 16 μs în densitate dublă, pentru discuri de 8 *inch*. 8257 are timp suficient pentru a prelua informațiile din RT, înainte ca acesta să fie reînscris (se previne apariția unei erori de ritm).

Condițiile de timp sînt mai restrictive la scriere, unde în același interval se fac două transferuri DMA. Pe canalul 2 se transferă simbolurile de informație, iar pe canalul 0 se transferă calitatea acestora (cu sau fără violare de cod). Transferul are loc întrețesut, declanșat de semnalul de delimitare a octeților, LODR, figura 3.23. Cererile de transfer DMA se fac pe rînd, mai întîi pe canalul 2, și după aceea pe canalul 0, astfel încît cele 2 circuite 8212 se încarcă, primul cu informația de codat, celălalt cu indicatorii, referitori la caracterul cu/fără violare a codării fiecărui simbol. Cererile de transfer se achită cu semnalele DACK0, 2, de la DMA. Transferurile au loc în 5 μs, interval suficient de mic pentru a nu apărea eroare de ritm. Validarea generală a transferurilor DMA în citire se face la apariția HRQ, sincronizarea pe începutul de bloc a cererii de transfer, DRQ, lansate prin program, ca în schema din figura 3.23. La scriere, validarea provine de la semnalul WRIT, lansat de asemenea prin program, întîrziat cu o semiperioadă a ceasului de referință de 2MHz, vezi § 3.4.3.6, transferurile începînd numai după ce se programează și registrul de comandă al 8257. Reprogramarea 8257, canalul 3, și funcționarea DMA în scriere și formatare se vor detalia la explicarea programelor aferente acestor operații. Încheierea transferurilor DMA este însoțită de semnalarea TC, *Terminal Count*, sfîrșit de transfer, care încarcă un registru 7495, decalat serie la fiecare semnal de delimitare octet, B7TIM,

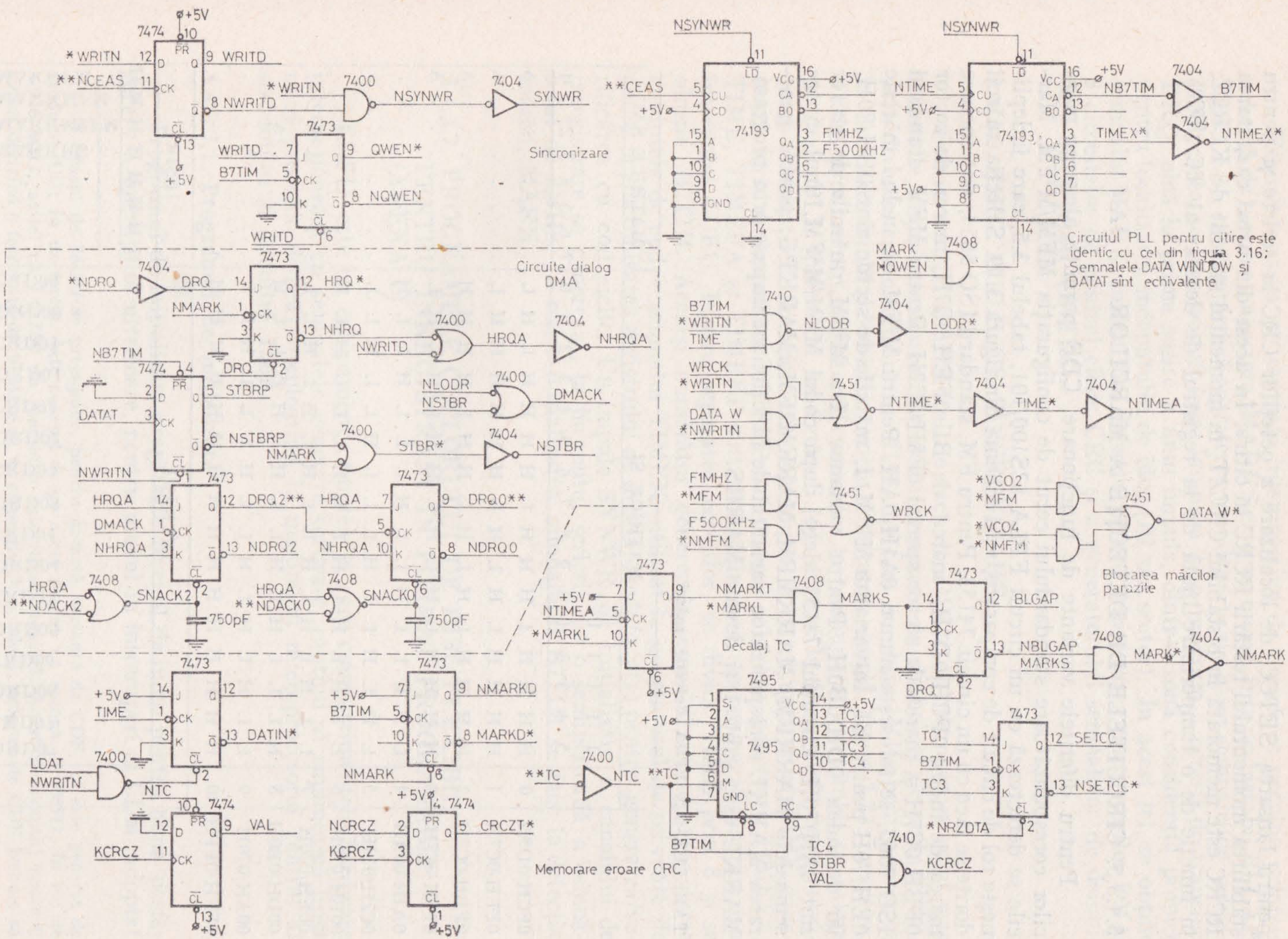


Fig. 3.23. CDFU. Dialog cu magistrala, circuitele de sincronizare, memorarea erorii CRC, blocarea mărcilor parazite, decalajul TC, generarea SETCC.

pentru lansarea SETCC, de localizare a octeților CRC la scriere și pentru stabilirea momentului testării RCRC în citire. În acest din urmă caz, starea RCRC este memorată în bistabilul CRCZT, în momentul stabilit de KCRCZ, în funcție de o temporizare luată de la registrul de decalaj al TC (TC4).

3.4.3.5. CIRCUITELE DE DETECȚIE A MĂRCILOR

Pentru diferitele variante de funcționare, CDF permite alegerea mărcilor corespunzătoare standardului cerut de configurația MFM/M²FM. Mărcile se detectează cu un circuit FPLA 82S100/101, tabelul 3.5, care îndeplinește rol de circuit de comparare, vezi schema din figura 3.19. Selecția mărcii dorite se face cu un circuit 74153. Pentru FM, standard ISO 5654, sînt incluse marca de index 0FCH/0D7H, marca de BI 0FEH/0C7H, marca de sector 0FBH/0C7H și marca de sector special 0F8H/0C7H. Pentru MFM, standard ISO 7065, se folosește numai 0A1H/0AH. Pentru M²FM, se includ mărcile 0EH/70H pentru BI. În varianta M²FM, I, mărcile de sector sînt 0BH/70H, iar de index 0DH/0B0H. Pentru varianta H a M²FM, mărcile de sector sînt 0AH/70H. Circuitul 74153 alege, după codul MFM/M²FM, unul dintre semnalele MARKFM, MARKMFM, MARKM2FM, MARKPG, pentru generarea MARKL corespunzător variantei de funcționare impuse prin program. MARKL este subțiat și devine MARKS.

TABELUL 3.5. **FPLA detecție mărci**

		MATRICE SI																MATRICE SAU							
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0FCH/0D7H	0	H	H	H	H	L	H	H	H	L	H	H	H	H	L	H	L	.	.	.	.	.	.	.	A
0FEH/0C7H	1	H	H	H	H	L	H	L	H	L	H	H	H	H	H	H	L	.	.	.	.	.	.	.	A
0FBH/0C7H	2	H	H	H	H	L	H	L	H	L	H	H	L	H	H	H	H	.	.	.	.	.	.	.	A
0F8H/0C7H	3	H	H	H	H	L	H	L	H	L	H	H	L	H	L	H	L	.	.	.	.	.	.	.	A
0A1H/00AH	4	L	H	L	L	L	H	L	L	H	L	L	L	H	L	L	H	.	.	.	.	.	.	A	.
0C2H/014H	5	L	H	L	H	L	L	H	L	L	L	H	L	L	H	L	L	.	.	.	.	.	.	A	.
00BH/070H	6	L	L	H	L	H	L	H	L	L	H	L	L	L	H	L	H	.	.	.	.	.	A	.	.
00EH/070H	7	L	L	H	L	H	L	H	L	L	H	L	H	L	H	L	L	.	.	.	.	A	A	.	.
00DH/0B0H	8	H	L	L	L	H	L	H	L	L	H	L	H	L	L	L	H	.	.	.	.	.	A	.	
00AH/070H	9	L	L	H	L	H	L	H	L	L	H	L	L	L	H	L	L	.	.	.	.	A	.	.	.
0FFH/0FFH	10	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	A	.	.	.	.	.	.	.
		INTRĂRI																IEȘIRI							
																		H	H	H	H	H	H	H	H
		DRD07	CRD07	DRD06	CRD06	DRD05	CRD05	DRD04	CRD04	DRD03	CRD03	DRD02	CRD02	DRD01	CRD01	DRD00	CRD00					MARKPG	MARKM2FM	MARKMFM	MARKFM

Resincronizarea pe semnale intempestive de marcă se previne printr-o schemă de blocare a acestora. MARKS este memorat într-un bistabil BLGAP, figura 3.23, care împiedică trimiterea mai departe și a altor semnale de marcă. Deblocarea se face numai între lansarea prin program a cererii de transfer, DRQ, pînă la prima sosire a unui impuls, dintr-o zonă considerată propice pentru localizarea începutului de bloc (BI sau sector). În acest fel, se obține semnalul MARK, folosit mai departe pentru inițializarea transferurilor DMA, a circuitelor de control ciclic CRC și a generatorului semnalelor de delimitare între octeți (B7TIM).

3.4.3.6. CIRCUITELE DE SINCRONIZARE

Pentru scriere, generatorul de tact de 2MHz* al microprocesorului este folosit la obținerea semnalelor de sincronizare pe simboluri de cod, TIME, prin divizare cu un numărător 74193, din care se obțin 1MHz pentru scrierea în densitate dublă și 500 kHz pentru densitatea simplă, figura 3.23. La citire, TIME provine din oscilatorul comandat în tensiune al circuitului cu calare pe fază, PLL (prezentat în figura 3.16), divizat la frecvențele corespunzătoare densității de înregistrare programate. Frontul urcător al TIME generează seria de impulsuri TIMEFP, iar frontul coborîtor generează TIMEFA. TIME, TIMEFA și TIMEFP sînt semnalele care sincronizează funcționarea întregului CDFU. Pentru separarea octeților, se divide TIME prin 8 și se obține B7TIM. Același numărător generează semnalul de separare între cuvintele de cod — semnalul TIMEX, utilizat la generatorul de secvențe de cod și la sincronizarea registrului de decalaj *modulo* $p(x)$ pentru generarea/verificarea cu cod ciclic. Un impuls SYNWR de inițializare a canalului de înregistrare se generează, la începutul scrierii, pe o semiperioadă a semnalului de 2MHz. În citire inițializarea semnalului B7TIM se face la detecția mărcii, MARK.

3.4.3.7. PROGRAMELE CUPLORULUI DE DISC FLEXIBIL PENTRU ÎNREGISTRĂRI ÎN FORMAT PROGRAMABIL (ANEXA C)

CDFU este un echipament complex în care compartimentele hardware interacționează cu semnalele generate software, căutînd să suplinească lipsa unui procesor dedicat, exploatînd la maximum microprocesorul central al sistemului și circuitul DMA.

3.4.3.7.1. **Programul de inițializare**

Programul de inițializare pregătește terenul de lucru pentru buna desfășurare a operațiilor ulterioare, pentru modul în întreruperi al CDM, adoptat

---

* Ceasul de 2MHz se presupune, pentru această variantă de CDF, că este generat de modulul UC; în exemplul dat, cuarțul de pe acest modul va rezona pe 4MHz, în loc de 4,916 MHz, astfel că UC-Z80 va funcționa sub viteza maximă. Acest dezavantaj poate fi evitat divizînd o frecvență de 10 MHz cu 4, respectiv 5. Evident, programele CDF trebuie să corespundă variantei de cuarț alese.

pentru toate variantele de CDF. În afara operațiilor uzuale, de pregătire a vectorilor de întrerupere, testarea stării READY în vederea stabilirii configurației SSDF și recalibrarea UDF-urilor, programul va trebui să determine modul de lucru al circuitului 8255 din figura 3.18. Pentru CDFU se alege 8255 în modul 0 de funcționare, cu *port*-urile A și C a cîte 8 ieșiri și cu *port*-ul B cu 8 intrări (cuvînt de control 82H). Circuitele de comandă la UDF sînt astfel proiectate încît, între aplicarea RESET-ului și programarea configurației, interval în care ieșirile sînt inactive, interfața UDF să se mențină inactivă și să nu apară semnale nedorite, periclitînd integritatea suportului magnetic, în timpul resetării. Prima operație care se efectuează după RESET înlătură nedeterminarea, trimițînd, în *port*-ul de control, configurația ce menține interfața inactivă (rutina PZFL).

Pentru pregătirea unor date necesare operațiilor de scriere și formatare, incluzînd configurații de mărci, structuri de BI etc., se copiază o serie de constante dintr-o zonă nemodificată în zona de lucru. Acestea vor fi prelucrate și transferate octet cu octet prin DMA spre circuitele de scriere. Pentru fiecare operație s-au definit adresele semnificative, în funcție de rolul pe care îl joacă în diferite etape ale operațiilor de înregistrare (scriere sau formatare).

Inițializarea continuă cu testul prezenței CDFU pe magistrală. La citirea unui port inexistent, *magistrala răspunde cu un octet nul*. Testul de zero, pe octetul citit din *port*-ul de control, stabilește prezența sau absența CDFU, care este codată conform cerințelor sistemului de operare, SO, în *port*-ul de stare (adresă 78H, poziția 3). În continuare se testează starea READY a fiecărui UDF și se completează pozițiile 0, 1 din *port*-ul de stare 78H.

Inițializarea se încheie cu recalibrarea fiecărui UDF și înscrierea 00H în locațiile care memorează numerele de pistă curentă.

3.4.3.7.2. Programul operației de căutare

Operația de căutare constă în deplasarea carului cu un număr de piste, dat de diferența între numărul de pistă curent și cel impus prin BP, vezi § 3.4.4.2, în sensul dat de semnul acestei diferențe, urmată de verificarea numărului de pistă la care s-a ajuns, prin citirea unui BI de pe această pistă.

După stabilirea parametrilor de lucru în densitate simplă/dublă, se realizează, cu rutina HDSP, deplasarea cu numărul de piste dorit, în sensul dorit, după o verificare prealabilă a corectitudinii ordinului lansat. Rutina compară numărul de pistă curent cu cel impus și alege sensul de deplasare chemînd subrutina UNPOT sau UNPIN, după caz, de cîte ori este nevoie. Impulsurile de STEP, lansate de 8255 prin program, se stabilesc de duratele impuse de UDF, realizate printr-o buclă de așteptare, cu subrutina WAIT. Verificarea se efectuează prin „încărcarea" capului (subrutina HLOD) și citirea unui BI, cu subrutina CAUTHB. Aceasta presupune citirea unui BI corect, fără eroare CRC, din cel mult 2×26 de încercări, echivalentul parcurgerii de două ori a unei piste. Dacă o astfel de citire este posibilă, se continuă cu verificarea numărului de pistă și se codifică eroarea, în cazul necoincidenței. Dacă BI nu se poate citi corect, verificarea este abandonată și se semnalează eroare de pistă neformatată în densitatea cu care s-a încercat citirea.

3.4.3.7.3. Programul operației de citire

Programul este conceput pentru a funcționa, cu aceleași rutine, în ambele densități de înregistrare — FM și MFM. De menționat că pentru formatele M^2FM sînt necesare pachete speciale de programe pentru densitatea dublă, deoarece întîrzierile impuse de dimensiunile specifice la BI și sector nu coincid cu întîrzierile din programele formatelor standard.

Principiul de lucru implică utilizarea continuă a circuitului DMA, care urmărește pista, citind cu transfer blocurile de identificare, urmate de citirea sectoarelor de date, fără transfer în memorie, pînă la găsirea BI impus prin BP; abia următorul sector se transferă efectiv în memoria principală. Păstrarea continuă a contactului intim cu structura formatului permite anticiparea zonei în care se așteaptă mărcile și crearea, prin temporizări software, a unor ferestre de așteptare a acestor mărci, care să evite mărcile detectate intempestiv. Zonele acestea pot fi destul de bine localizate, înaintea zonei de sincronizare din fața mărcilor valide. Manevrînd ca semnalul DRQ să inițializeze bistabilul BLGAP, figura 3.23, înainte de sosirea mărcii valide, se asigură ca acesta să fie primul impuls și unicul, care să poată trece spre circuitele de sincronizare, pînă la încheierea BI sau a sectorului ulterior, după care aceeași manevră a DRQ poate relua procesul. Pentru prima citire de BI se permite o toleranță destul de mare a numărului de încercări (2 ×26), deoarece trebuie să fie permisă sincronizarea cu formatul pistei. Odată sincronizarea realizată, BI și sectoarele următoare se vor citi în lanț cu transfer în memorie numai al sectoarelor cerute de BP. După fiecare sector citit, se examinează erorile posibile și se completează locația de eroare aferentă sectorului dintr-un tabel cu locații rezervate pentru fiecare sector. Se examinează prezența mărcii speciale 0F8H, absența mărcii de sector, într-o fereastră de timp anumită, stabilită în raport cu localizarea BI, eroarea la verificarea CRC a sectorului și absența unui BI corect cu numărul de ordine al sectorului căutat. Erorile se stabilesc examinînd semnalele CRCZ de la RCRC, HRQ de la bistabilul pregătit de DRQ, dar anclanșat la sosirea mărcii, și locațiile rezervate la care se transferă mărcile. DMA lucrează pe canalul 2, cu canalul 3 în *autoload*, pe care îl reînscrie după ce precedenta încărcare s-a transferat, eveniment semnalat prin testarea emisiei TC. Testul de TC servește și la evidențierea trecerii octeților CRC și determină momentul testării bistabilului CRCZ. Din această cauză se impune lansarea TC în a inte de cei doi octeți de control, pentru inițializarea circuitelor de memorare a stării RCRC. Pentru BI, DMA transferă, în zona rezervată, marca (mărcile, în MFM) și cei 4 octeți, emite TC, transferînd și cei 2 octeți CRC. Pentru sector se transferă în prealabil marca (mărcile), în zona rezervată, apoi conținutul util ajunge, prin cicli DMA, în memoria principală, după care, din nou, transferurile CRC se fac în zona rezervată. Se menține opțiunea cu verificare *software* a controlului ciclic, deși timpul de execuție al rutinei respective mărește mult valoarea timpului de acces la sector, cu avantajul eliminării RCRC. Pentru a se preveni „descărcarea" capului la citiri succesive de sectoare pe formate standard cu ordine aleatoare dezavantajoasă, s-a inclus în rutina de sincronizare pe BI și operația de „reîncărcare" a capului.

3.4.3.7.4. **Programul operației de scriere**

Principiul de funcționare corespunde conlucrării a două canale DMA, canalul 2 funcționînd în mod *autoload*, rezervat transferurilor informației de codat, iar canalul 0 transferînd indicatorii pentru violarea de cod, specifică zonei de înregistrare a mărcii de început de sector. După trecerea octeților de marcă, pe canalul 0 se restabilește modul de lucru fără violare de cod și transferurile se pot opri, preluarea automată din RT găsind același octet de validare la toate transferurile ulterioare. Funcționarea canalului 2 în *autoload* este impusă de necesitatea de a prelua, din zona rezervată, configurațiile specifice formatului — zona de sincronizare și marca — și de a continua neîntrerupt transferurile cu datele de înregistrat, revenind la constanta (00H sau 0FFH) specifică octetului care încheie înscrierea, după înregistrarea CRC.

Căutarea BI al sectorului ce urmează să fie înregistrat se face cu același algoritm ca și la citire, prin identificarea BI și transferul DMA în mod *verify*, fără transfer efectiv în memorie pentru sectoarele depășite. Cînd BI conține numărul de ordine al sectorului dorit, se continuă ciclurile DMA cu *autoload* în intervalul BI—sector, pînă la atingerea zonei de sincronizare. Se lansează comanda WRIT pe *port*-ul 8255, figura 3.18, și se transferă din zona rezervată datele pentru scriere, octet cu octet, destinate zonei de sincronizare și mărcii de început de sector. Pe canalul 0 octetul corespunzător mărcii, conține configurația violării de cod. Odată încheiat transferul mărcii pe canalul 2, registrele sale se încarcă, prin *autoload*, cu conținutul înregistrat în registrele canalului 3, unde au fost pregătite adresa și numărul de octeți pentru transferul din memoria principală. Pe canalul 0 se transferă anularea violării. La sfîrșitul operației de scriere a datelor din memoria principală se sesizează apariția TC de la DMA, pentru pregătirea încheierii scrierii, prin completarea cu octeții de verificare CRC, urmați de octetul constant final.

Deoarece, după trecerea mărcii, următoarele cuvinte de cod nu mai conțin violări, transferurile pe canalul 0 pot înceta, odată cu încărcarea primului octet după marcă. Încărcarea în RSD a indicatorilor de violare continuă, automat, prin preluarea aceluiași octet din 8212 (*RT scriere* în figura 3.19).

Comutarea canalului de scriere pe ieșirea serială NQ15 a RCRC se face după detecția TC de la sfîrșitul de bloc, figura 3.20. Pentru a se decela acest TC față de celelalte, impuse de modul de lucru în *autoload*, se formează separat un semnal NRZDTA, care permite generarea, pe următorii 2 octeți de CRC, a semnalului SETCC, figura 3.23. Codificatorul de secvență alege, dacă SETCC = 1, secvența serie de la ieșirea RCRC în locul celei sosite de la RSD. (§ 3.4.3.2.)

Scrierea impune timpi de sincronizare foarte preciși între diferite faze. Microprocesorului îi este impusă frecvența de ceas de 2 MHz, sau alt multiplu exact al vitezei de transfer cu discul, de 250 kHz (densitate simplă) sau 500 kHz (densitate dublă). Buclele de întîrziere sînt introduse pentru a compensa decalajele generate de parcurgerea programului. O primă buclă reglează intervalul de timp de la detecția BI corect și pînă la lansarea comenzii de scriere, pentru ca noul sector înscris să se suprapună peste cel vechi, în toleranțe acceptabile.

Trecerea TC de după zona de constante — sincronizare și marcă — este testată și permite reprogramarea canalului 3, pregătindu-l pentru sfîrșitul de bloc. După detecția și a TC de la sfîrșitul datelor înscrise în bloc, se așteaptă trecerea CRC și se dezactivează NRZDTA, DRQ, WRIT, întrerupîndu-se transferurile DMA.

Transferul unui bloc este inclus într-o buclă, LOOPA, în care se pot transfera mai multe sectoare, pînă la o pistă, cu resincronizare, de fiecare dată, pe BI corespunzător. De notat că, la orice lectură de BI, în scriere sau citire, se face și o programare a canalului 0 pentru a asigura o eventuală scriere. Canalul 0 nu este activat decît la comanda WRIT, lansată numai în comenzi de înregistrare.

3.4.3.7.5. Programul operației de formatare

Procedura de la scriere este respectată și la formatare numai că, de această dată, se înregistrează toată pista, inclusiv BI pentru fiecare sector. Ordinea de înscriere a BI poate fi normală (1, 2, 3, 4 ...) sau aleatoare, în funcție de un tabel specificat în BP. În prealabil se verifică validitatea numărului de pistă și dacă UDF nu este protejată, semnalîndu-se eventualele erori. În cazul desfășurării normale, se completează un tabel, din zona rezervată, cu succesiunea dorită a numerelor de sector. La formatare aleatoare, în tabel se copiază ordinea furnizată la adresa plasată în BP, octeții 6, 7. Se efectuează deplasarea și „încărcarea" capului la pista dorită. În cazul formatării pistei 0 se verifică semnalul TRACK 00.

Operația propriu-zisă de transfer de date începe după ce a fost pregătită zona rezervată, conform configurației începutului de pistă, imediat după ce s-a întîlnit semnalul INDEX. Se lansează WRIT pe un *port* al circuitului 8255. WRIT va fi activ pe toată durata formatării. Conținutul înregistrării se va determina manevrînd numai DRQ și NRZDTA, cu testul TC și încărcarea registrelor DMA de pe canalele 0, 2 și 3. Bucle de temporizare precis calculate introduc valorile impuse pentru intervalele INDEX-BI, BI—sector, sector—BI și pentru lungimea sectorului.

În timpul scrierii unui sector, se modifică în zona rezervată numărul de ordine al BI următor, conform tabelului completat mai înainte. DRQ este manevrat pentru pregătirea reinițializării RCRC, care trebuie să calculeze octeții de control ciclic la fiecare BI și sector ale pistei care se formatează. Deoarece momentele de timp ale programării DMA sînt fixate, pentru a nu perturba transferurile în curs, se folosesc numai bucle de timp sincronizate cu microprocesorul. Modulul UC furnizează concomitent și semnalul cu frecvența de referință a scrierii, 2MHz. După înscrierea ultimului sector, intervalul final se completează, conform formatului standard, pînă la noua întîlnire a impulsului de INDEX.

Programul de formatare are două variante, FORSD și FORDD, pentru standardul în densitate simplă, FM, și în densitate dublă, MFM. Procedurile se aseamănă, cu modificări date de frecvența de transfer și particularitățile de format, în special în zona de mărci. Buclele de întîrziere realizate prin program stabilesc valorile specifice ale intervalelor dintre reprogramările canalelor DMA. Constantele din zona rezervată vor corespunde configurațiilor de mărci și octeți de „umplutură" din intervaluri (*gap*-uri).

3.4.4. CIRCUITELE DE DIALOG CU MAGISTRALA

3.4.4.1. PREZENTAREA CIRCUITELOR

Toate cele trei module CDF – cu LSI 8271, cu LSI 8272 și cuplorul programabil, CDFU, au un set de circuite și programe asemănătoare, necesare pentru comunicarea între FRM specializat și memoria microcaculatorului „gazdă". Acestea sînt impuse de faptul că același sistem de operare trebuie să lucreze cu suportul magnetic la fel, indiferent de alegerea CDF. Circuitele și programele sînt concepute pentru interacțiunea compartimentului sistemului de operare specializat în gestiunea fișierelor de pe discul flexibil și setul de circuite și programe specifice fiecărui cuplor. Microcalculatorul pe 8 biți din exemplul nostru este compatibil cu sistemele de operare ISIS-II și CP/M. De fapt, compatibilitatea cu sistemul de operare ISIS-II este suficientă, pentru că există o variantă adaptată a sistemului CP/M, care lucrează în același mod cu SSDF. Sistemul modular este compatibil și cu variantele SFDX-II și CP/M implementate pe microcaculatoarele din seria M-18, M-118, cu echipamentele TPD, JUNIOR și MS-100, fabricate în România.

Procedurile legate de disc ale sistemului de operare ISIS-II corespund unui SSDF inteligent, condus de unitatea centrală a microcalculatorului prin *port*-urile aflate la adresele 78H÷7FH (pentru primul set de 2 UDF), *port*-uri care mediază schimbul de comenzi/stări. Transferurile de date au loc la inițiativa SSDF, prin cicli DMA. În prealabil, SO completează, în memoria principală a microcalculatorului, un vector de comandă, *Input/Output Parameter Block*, BP, cu parametrii și codul operației de efectuat. Adresa acestui BP este trimisă, la SSDF, în 2 octeți succesivi, prin *port*-urile de adresă 079H și 07AH. SSDF sesizează înscrierea octetului cel mai semnificativ în *port*-ul de adresă 07AH, citește BP de la adresa aflată și execută operația comandată. În timpul lucrului, SSDF este în stare „ocupat", comunicată spre microcalculator prin intermediul *port*-ului comun de adresă 078H. Unitatea centrală testează eliberarea SSDF, pentru cercetarea modului de încheiere a operației, cu sau fără eroare, și îi trimite ordinul următor. În caz de eroare, microcalculatorul află natura acesteia, așa cum ea a fost codificată de SSDF, în *port*-ul de adresă 7BH. Starea SSDF este sintetizată în *port*-ul 078H.

CDM simulează existența cuplorului inteligent, folosind întreruperi generate hardware, ca în schema din figura 3.24. Microprocesorul sistemului substituie unitatea centrală proprie a cuplorului inteligent, pe timpul execuției ordinelor de disc. Se obține un coeficient maxim de utilizare a unității centrale, care ar fi rămas, în acest interval de timp, într-o buclă de testare a încheierii operației cu discul. Soluția aleasă în exemplu corespunde generării unei întreruperi *de execuție* la fiecare lansare a unei noi comenzi pentru SSDF, în care să se realizeze substituția microprocesorului. La fiecare operație de aflare a codului erorii se lansează o altă întrerupere *de achitare*. Automatul de generare a întreruperilor este manevrat de accesele la *port*-urile 078H÷÷07FH, după cum urmează: înscrierea în 07AH generează INT6, citirea ulterioară din 07BH lansează INT5. Automatul este inițializat la înscrierea adresei BP în *port*-urile 079H, 07AH și la RESET.

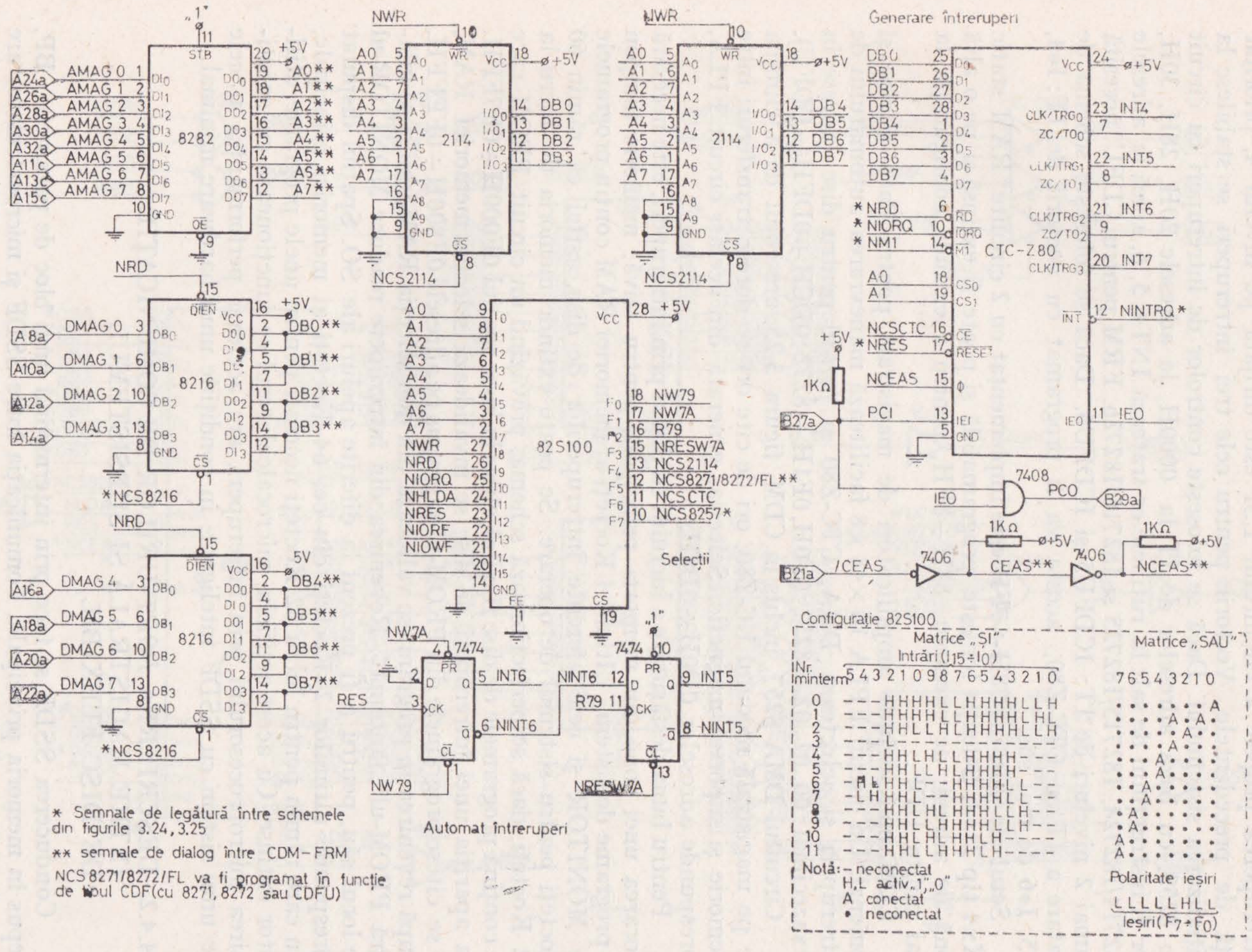


Fig. 3.24. **Circuitele de dialog cu magistrala microsistemului, CDM.** ***Buffer*-e date/adrese/comenzi, selecții, generare întreruperi, automat întreruperi.**

Rutina *de execuție* este apelată la nivelul de întrerupere 6, iar cea *de achitare* este stabilită pe nivelul de întrerupere 5. Pentru FRM cu 8271/72, întreruperea proprie a circuitului LSI este dirijată pe nivelul 4, prioritar față de precedentele. Vectorii pentru cele trei întreruperi se stabilesc la inițializarea sistemului. Dacă se folosește controlor de întreruperi cu circuit tip 8259, cu baza adreselor de IT la 0000H, la adresele 20H, 28H, 30H, vor fi instrucțiuni de salt la rutinele ce tratează INT4, 5, 6, adică la adresele I82714/I82724, I82715/I82725 și I82716/I82726. FRM pentru CDFU necesită numai 2 niveluri de IT: ICDFU5 și ICDFU6. Dacă se folosește schema de tratare a IT cu CTC-Z80, acesta va fi programat cu vectorii de IT: I*4, I*5, I*6 de mai sus.

Setul de *port*-uri 78H÷7FH este implementat cu 2 circuite RAM static 1Kx4 tip 2114. Selecția lor este programabilă și poate fi extinsă și pe o altă zonă de adrese, de exemplu, 88H÷8FH, pentru a extinde configurația la mai multe UDF.

CDM mai include amplificatori de magistrală pentru date, adrese și comenzi. Un circuit FPLA 16 ×48 ×8 facilitează manevrarea automatului de întreruperi și selectează DMA, CTC-Z80 și *port*-ul pentru disc (adrese în hexazecimal: 00, 01, 02 — 8271; 0F0H, 0F1H — 8272; 0DCH÷0DFH — CDFU).

Circuitul DMA 8257, inclus în CDM, figura 3.25, are rolul de a elimina de pe magistrală modulul UC-Z80, ori de cîte ori se doresc transferuri între memorie și suportul magnetic. Selecția, generată din același circuit FPLA, corespunde adreselor 0D0H÷0D8H.

Pentru buna desfășurare a lucrului cu discul prin întreruperi, este necesară alocarea unei zone din memoria principală pentru stivă, memorie tampon și programe de sistem. Ultimii 4 Kocteți ai memoriei RAM conțin programele de MONITOR și cele aferente întreruperilor de disc, astfel că rămîn 60 Kocteți pentru sistemul de operare. Se poate extinde memoria de lucru la 62 Kocteți, dacă se perfecționează schema, prevăzînd un circuit PROM care să conțină programele de disc flexibil, eliberîndu-se spațiul 0F000H÷0F7FFH. La apariția unei întreruperi de disc se invalidează selecția memoriei RAM, și se citesc programele din PROM-ul suprapus pe adresele 0F800H÷0FFFFH. După revenirea în program, se validează din nou selecția RAM-ului și se înlătură PROM-ul „fantomă". Revenirea din întrerupere readuce MONITOR-ul pe locul său, pentru a fi prezent la diferite apeluri ale SO. Spațiul disputat corespunde ultimilor 2 Kocteți din cei 64 Kocteți ai memoriei principale, din care rămîn pentru lucru 62 Kocteți necesari pentru unele programe utilizator extinse. Cu acest artificiu, microcalculatorul ce funcționează cu substituirea microprocesorului prin întreruperi, are aceleași performanțe cu cele ale unui sistem cu SSDF inteligent, în condițiile unui hardware minimal.

3.4.4.2. DESCRIEREA *PORT*-URILOR DE COMUNICAȚIE ÎNTRE MAGISTRALĂ ȘI SUBSISTEMUL DE DISC FLEXIBIL

Conducerea SSDF se face prin intermediul unui bloc de parametri, BP, depus în memoria principală. Comunicația între SSDF și microcalculator are loc prin intermediul unor *port*-uri pentru transferul adresei BP și a stărilor SSDF. Blocurile de date — conținutul sectoarelor de pe disc — se transferă prin acces direct la memorie, DMA.

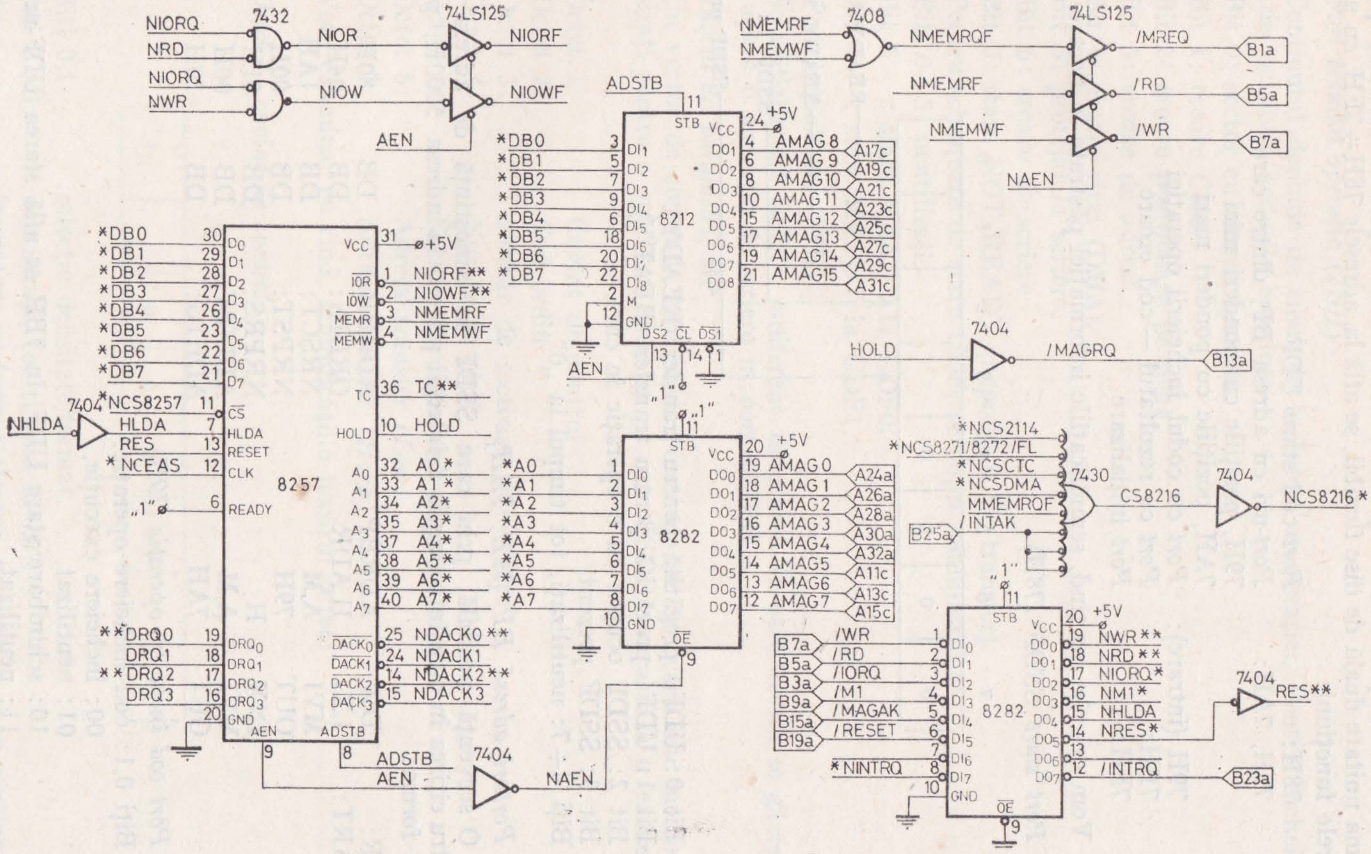


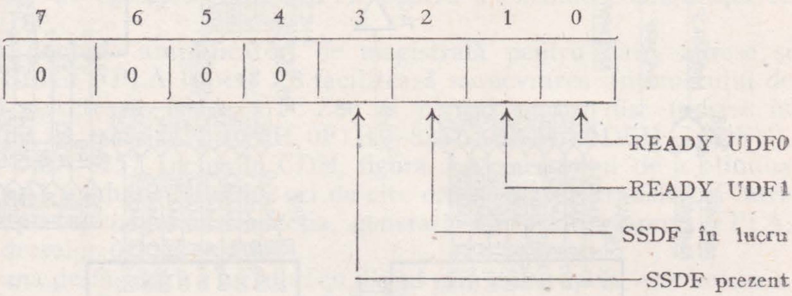
Fig. 3.25. CDM. Circuitul DMA, *port*-uri de comunicaţie cu magistrala.

Un set de *port*-uri, acceptat de sistemul de operare ISIS-II, pentru prima unitate duală de disc flexibil, se află la adresele 78H÷7FH, cu următoarele funcţiuni:

78H:	*Port* cod stare UDF
79H, 7AH:	*Port*-uri cu adresa BP, dintre care:
	79H, poziţiile cu ponderi mici
	7AH, poziţiile cu ponderi mari
79H (intrare):	*Port* cu codul încheierii operaţiei
7BH:	*Port* cu rezultatul — cod eroare
7FH:	*Port* iniţializare

Vom descrie, pe rînd, semnificaţiile informaţiei păstrate în fiecare *port*.

Port stare SSDF (78H)



Bit 0: UDF0 pregătită pentru transfer (READY).
Bit 1: UDF1 pregătită pentru transfer (READY).
Bit 2: SSDF ocupat cu o operaţie în curs.
Bit 3: SSDF prezent.
Biţi 4÷7: neutilizaţi, tot timpul la „0".

Porturi adresă BP (79H, 7AH)

O secvenţă posibilă, prin care SSDF ia cunoştinţă de adresa BP, pentru citirea în densitate simplă a întregii piste 0 la adresa 3000H, poate avea forma:

```
ADR:    DW   ADRBP       ADRBP:   DB   80H
START:  LXI  H,ADR       ORDIN:   DB   04H
        MVI  A,M         NRSCT:   DB   1AH
        OUT  79H         NRPST:   DB   00H
        INX  H           NRPRS:   DB   01H
        MVI  A,M         ADLOW:   DB   00H
        OUT  7AH         ADHIG:   DB   30H
```

Port cod încheiere operaţie (79H)

Biţi 0,1: cod încheiere operaţie:
- 00: încheiere execuţie,
- 01: neutilizat,
- 10: schimbare stare UDF, în 7BH se află starea UDF-urilor,
- 11: neutilizat.

Biţi 2÷7: neutilizaţi, tot timpul la „0".

Port rezultat operaţie (*7BH*)

Conţinutul depinde de tipul de încheiere a operaţiei. Pentru încheiere operaţie, se obţin codurile erorilor:

Bit 0: sector cu marcă specială.
Bit 1: eroare CRC.
Bit 2: eroare poziţionare pe pistă.
Bit 3: eroare de adresă.
Bit 4: eroare ritm (DMA).
Bit 5: protejat la scriere.
Bit 6: eroare de scriere.
Bit 7: stare NOT READY (nepregătit de transfer).

Pentru schimbare de stare UDF, se obţin semnificaţiile:

Biţi 0÷5: neutilizaţi.
Bit 6 : stare READY la UDF0.
Bit 7 : stare READY la UDF1.

Port iniţializare (7FH)

Se iniţializează SSDF, indiferent de starea în care acesta se găseşte, dacă apare o operaţie de scriere în *port*-ul 7FH.

Blocul de parametri, BP

Un vector de comandă este format dintr-un şir de octeţi din memorie, organizaţi pentru a specifica operaţia pe care SSDF urmează să o execute:

Octet 1:	Cuvînt de control
Octet 2:	Cod ordin
Octet 3:	Număr de sectoare transferate
Octet 4:	Număr de pistă
Octet 5:	Număr de ordine al primului sector
Octet 6:	Adresa transfer *(Low)*
Octet 7:	Adresa transfer *(High)*

Vom examina, pe rînd, semnificaţiile fiecărui octet:

Octet 1: *cuvînt de control*

7	6	5	4	3	2	1	0
1		0	0	0	0		

Biţi 0,1: selecţie densitate/format.
Biţi 2÷5: neutilizaţi, tot timpul la „0".
Bit 6: formatare cu format aleator, în ordinea şi cu conţinutul de bloc specificate la adresa din octeţii 6, 7 ai BP.
Bit 7: tot timpul la „1".

Octet 2: cod ordin

Biții 0÷2: cod al operației:
000 neutilizat,
001 căutare,
010 formatare,
011 recalibrare,
100 citire,
101 citire cu verificare,
110 scriere,
111 scriere specială.

Bit 3: rezervat, tot timpul la „0".

Biții 4,5: 00 selecție UDF0,
11 selecție UDF1.

Biții 6,7: rezervați, tot timpul la „0".

Octet 3: număr de sectoare

Se specifică numărul de sectoare pe care se face transferul (1÷26). Octet nesemnificativ la operațiile de căutare și recalibrare.

Octet 4: numărul de pistă

Se specifică numărul de ordine al pistei la care se referă operația (0÷76). Nu este luat în considerare la ordinul de recalibrare.

Octet 5: numărul primului sector

Se specifică numărul de ordine al primului sector cu care începe transferul (1÷26). Sistemul de operare ISIS-II semnalează adresa unității de disc pe poziția 5, astfel că, pentru UDF1, sectoarele se numerotează cu valorile 21H÷3AH. Se asigură compatibilitatea cu SSDF lucrînd în densitate dublă, în mod M^2FM. Octetul 5 este irelevant la ordinele de recalibrare și căutare.

Octeții 6, 7: adresa transferului de date.

Se specifică adresa, din memoria principală, cu care se încep transferurile de date. Octetul 6 conține cele 8 poziții mai puțin semnificative. La ordinul de formatare aleatoare, octeții 6, 7 conțin adresa, din memoria principală, la care SO a completat un tabel cu ordinea de succesiune a sectoarelor.

3.4.4.3. PREZENTAREA PROGRAMELOR (ANEXELE A, B, C)

Ca și modulul CDM hardware, programele aferente sînt comune tuturor tipurilor de FRM prezentate.

Lucrul prin întreruperi cere ca rutinele SSDF să nu interfere cu programele din SO, a căror execuție o suspendă. Din această cauză, intrarea și ieșirea din aceste rutine se face cu salvarea și restaurarea tuturor registrelor microprocesorului, iar execuția rutinelor se face cu stivă separată, în afara zonei de memorie accesibile SO. În acest fel se evită periclitarea cursivității programului principal, singura deosebire față de funcționarea cu SSDF inteligent fiind aceea că microprocesorul nu mai testează, în buclă, încheierea operației comandate.

Programul aferent CDM operează cu *port*-urile prin care SO lucrează cu SSDF și cu rutinele specifice FRM atașat. În prealabil, se preia adresa BP, se copiază acesta din memoria principală în zona afectată CDF și se prelucrează pentru aflarea operației de efectuat și a adresei UDF căreia îi este destinată. Dintr-un tabel, în funcție de codul operației, se preia adresa rutinei corespunzătoare și se execută. Revenirea din rutină este urmată de preluarea rezultatului și stabilirea conținutului *port*-urilor 78H, 79H, 7BH conform codurilor de stare și eroare, astfel ca SO să decidă continuarea, în funcție de încheierea comenzii lansate anterior. La sfîrșitul execuției unei operații, SSDF este „blocat" („1" în poziția 2 din *port*-ul 78H) și se eliberează după testarea modului de încheiere. S-a menținut manevrarea acestui bit pentru compatibilitatea cu programele referitoare la gestiunea discului flexibil din SO, în care mai întîi se testează încheierea lucrului SSDF și apoi se preia octetul de mod al încheierii (*port* 79H) și cel al codului erorii (*port* 7BH). Următoarea operație se lansează numai *după* testarea eliberării SSDF („0" în poziția 2 din 78H).

3.5. SISTEME DE OPERARE PE DISC FLEXIBIL

Evoluția spectaculoasă a microcalculatoarelor s-a datorat, în mare măsură, facilităților oferite utilizatorilor de tehnică de calcul, care s-au văzut dotați cu posibilitatea de a transfera aplicații, de la sistemele de calcul mari, la microsistemele dedicate proprii, cu preț de cost scăzut, fără pierdere de performanță, cu un coeficient de accesibilitate mare. Pe lîngă microprocesor, principalul motor al acestei evoluții, discul flexibil, manevrat de un sistem de operare adecvat, a oferit posibilitatea implementării unor aplicații complexe, la viteze de lucru convenabile.

Sistemele de operare pe disc flexibil au fost dezvoltate după modelul celor corespunzătoare sistemelor de calcul mari, echipate cu discuri de capacitate mare și avînd viteză ridicată de lucru.

Sistemul de operare asistă utilizatorul în gestionarea resurselor sistemului (unitate centrală, memorie, periferice). Pentru a-și îndeplini funcțiile, SO presupune standardizarea transferurilor între resurse sub forma unor unități logice numite *fișiere*. Operațiile oferite de sistemele de operare se referă la diferite prelucrări asupra fișierelor. SO lansează comenzi pentru periferice, în conformitate cu prelucrările logice asupra fișierelor. În particular, pentru discul flexibil, se realizează corespondența între structura logică a fișierelor și organizarea înregistrărilor fizice pe suportul magnetic, gestiunea acestora în conformitate cu prelucrările logice efectuate.

O unitate duală de disc flexibil oferă o memorie externă de 0,5 Mocteți în densitate simplă, pe o suprafață, și 1 Moctet în densitate dublă pe o suprafață. Integrarea în sistem presupune realizarea mai multor niveluri de cuplare.

Un prim nivel, cel mai apropiat de natura fizică a înregistrării, constă într-un set de circuite electronice pentru conectarea între interfața microsistemului pe 8 biți și o interfață specifică perifericului, cu semnale de comandă/stare și transfer serial al datelor. Acest nivel poate fi asigurat de un modul FRM, realizat în jurul unui circuit LSI (8271, 8272) sau în tehnologie

MSI programabilă (FPLA), cu anumite circuite anexe (PLL, *buffere* etc.). Acest nivel a fost prezentat în § 2.3.1÷2.3.2, § 2.4.1÷2.4.2 şi §3.4.3.1÷3.4.3.6.

Al doilea nivel constă în programe specifice circuitului LSI sau logicii programate pentru îndeplinirea diferitelor ordine ca: scriere, citire, recalibrare etc., cu examinarea condiţiilor de încheiere a acestor operaţii. Acest nivel se referă la proceduri direct legate de particularităţile circuitului utilizat, care trebuie programat şi condus pentru realizarea operaţiilor specificate. Acest nivel a fost prezentat în § 2.3.3÷2.3.5, § 2.4.3÷2.4.6 şi § 3.4.3.7.

Al treilea nivel stabileşte un mod organizat de comandă şi control pentru discul flexibil, prin proceduri înţelese de sistemul de operare, care permit o conducere unitară şi clară a perifericului, folosind un vector de comandă şi examinînd un vector de rezultat. Acest nivel este realizat prin programe de transfer al comenzilor unitare în comenzi realizate de nivelul inferior. Acest nivel a fost acoperit de § 3.4.4.

Sistemul de operare propriu-zis, ales, de regulă, dintre cele uzuale, pentru facilitarea interschimbului de programe, corespunde nivelului superior, al patrulea, care transferă setul de macrocomenzi ale operaţiilor cu fişierele de date, în seturi de comenzi unitare specifice discului flexibil. Aceste comenzi vor fi prezentate în § 3.5.1.2. pentru ISIS-II şi § 3.5.2. pentru CP/M.

Setul propriu de comenzi al circuitelor LSI specializate diferă de setul unitar cerut de sistemul de operare, astfel că nivelul trei este destul de voluminos. Într-o implementare cu logică programabilă, se pot alege ca proceduri pentru circuitele realizate chiar cele specifice setului unitar cerut de sistemul de operare, contopind astfel nivelul 2 şi 3 într-unul singur.

3.5.1. SISTEMUL DE OPERARE ISIS-II

Elaborat pentru primele sisteme de dezvoltare tip MDS — *Microcomputer Development System* — sistemul de operare ISIS-II foloseşte unităţi de disc duale, în densitate simplă, format IBM şi poate lucra cu microprocesoare compatibile cu 8080, cu memorie între 32÷62 Kocteţi, cu organizare ce va fi prezentată la § 3.5.3.1. Sub acest sistem se pot rula aplicaţii dedicate, folosind discul pentru culegere şi prelucrare de date. Ca sistem de calcul, permite dezvoltarea de programe complexe pentru o gamă largă de microprocesoare şi permite lucrul în limbaje evoluate BASIC, FORTRAN, COBOL, PASCAL etc.

3.5.1.1. ORGANIZAREA DISCHETEI

Prima pistă este dedicată sistemului de operare propriu-zis, care se încarcă între locaţiile 40H÷2FFFH, cu memorie tampon la 3000H÷3680H.

Primele piste conţin fişierele ISIS.TO, ISIS.LAB, ISIS.DIR, ISIS.MAP, proprii gestiunii blocurilor dischetei.

ISIS.TO conţine un program de încărcare a sistemului de operare. ISIS.LAB conţine titlul dischetei şi factorul de întreţesere la formatarea fiecăreia dintre cele 77 piste.

ISIS.DIR conține, în grupe de 26 octeți, numele și localizarea fiecărui fișier, împreună cu caracteristicile fiecăruia.

ISIS.MAP este o hartă a discului cu blocurile fizice libere și ocupate, explorată la fiecare operație de creare, modificare sau ștergere a fișierelor.

Un fișier este organizat pe baza unui tabel al sectoarelor componente, aflat în primul sector adresat în ISIS.DIR. În cazul în care un singur bloc nu este suficient pentru tabel, se pot înlănțui mai multe blocuri „tabel". Un bloc „tabel" conține adresa blocului „tabel" precedent (primii 2 octeți) și ai blocului „tabel" următor (următorii 2 octeți). Dacă nu mai există succesori sau predecesori, octeții corespunzători sînt nuli.

Discul sistem mai conține în plus fișierele ISIS.BIN și ISIS.CLI, care includ sistemul de operare. O dischetă, în densitate simplă, conține pe cele 77 piste, a cîte 26 sectoare, un număr de 2 002 sectoare a cîte 128 octeți de date, totalizînd o capacitate utilă de 256 256 octeți (250 Kocteți).

3.5.1.2. RUTINELE DE SISTEM ALE ISIS-II

Sistemul de operare este încărcat în memorie în zona de adrese 40H÷÷2FFFH și folosește ca memorie tampon zona 3000H÷3680H. În spațiul rămas, pînă la zona superioară a memoriei, se pot rula programele de aplicații. Sistemul de operare posedă un set de 15 rutine de sistem, pe baza cărora se construiesc comenzi de sistem și programe dedicate unor aplicații. Fiecare rutină are o zonă de parametri, completată de utilizator.

Cele 15 rutine de sistem sînt următoarele:

Cod 0: *OPEN*: deschiderea fișierului; un fișier specificat este conectat la o conductă specificată a sistemului de operare.

Cod 1, *CLOSE*: închiderea fișierului; o conductă specificată este deconectată de la fișierul pe care acesta a fost în prealabil deschis.

Cod 2, *DELETE*: ștergere fișier; eliminarea de pe suportul magnetic a fișierului specificat, cu eliberarea înregistrărilor aferente, devenite disponibile pentru alte fișiere.

Cod 3, *READ*: citire din fișier; se transferă date dintr-o conductă, pe care este deschis un fișier, într-o zonă specificată de memorie, pe o lungime de octeți specificată, cu actualizarea unui *marker* de octet.

Cod 4, *WRITE*: scriere în fișier; se transferă date, dintr-o zonă de memorie specificată, în fișierul deschis pe o conductă specificată, pe o lungime de octeți specificată, cu actualizarea *marker*-ului de octet.

Cod 5, *SEEK*: caută *marker*; se află sau schimbă valoarea numărului de ordine ce desemnează octetul curent, aflat înainte de a fi transferat pe conducta specificată (*marker* de octet).

Cod 6, *LOAD:* încarcă fișier; se depune în memorie fișierul specificat, cu un decalaj specificată și se transferă controlul la o adresă specificată.

Cod 7, *RENAME*: schimbare de nume fișier; se înlocuiește, în fișierul ISIS.DIR, numele fișierului specificat, cu un altul.

Cod 8, *CONSOL*: transferă consola; se înlocuiește consola sistemului cu două fișiere, care vor corespunde ieșirilor pe display și intrărilor de la tastatură; cele două fișiere vor conduce în continuare sistemul.

Cod 9, *EXIT*: ieșire în ISIS-II; se transferă controlul, din programul utilizator, sistemului de operare.

Cod 10, *ATTRIB*: schimbă atribute de fişier; se modifică atributele ISWF ale fişierului specific (I — invizibil, S — sistem, W — protejat la scriere, F — format).

Cod 11, *RESCAN*: poziţionare la început de linie; se aduce *marker*-ul de octet la început de linie, într-o zonă tampon specifică editării, în care s-a citit în prealabil dintr-un fişier deschis.

Cod 12, *ERROR*: semnalare a erorii; se trimite la consolă mesajul de eroare.

Cod 13, *WHOCON*: caută consola; se determină fişierul asignat drept consolă (intrare sau ieşire).

Cod 14, *SPATH*: caută fişierul; se află informaţii asupra fişierului specificat (de exemplu, pe ce periferic este rezident).

Cu ajutorul acestor 15 rutine de sistem se pot construi programe care să înglobeze prelucrări de fişiere. ISIS-II permite coexistenţa a 6 fişiere deschise simultan, în afara fişierelor :CI: şi :CO:, continuu deschise.

3.5.2. SISTEMUL DE OPERARE CP/M

Elaborat de *Digital Research* pentru a pune în valoare minicalculatoarele cu microprocesor 8080 şi disc flexibil, sistemul de operare CP/M are, ca principal ascendent, asupra celorlalte SO, larga interschimbabilitate a fişierelor pe suport magnetic. Această proprietate este generată de separarea zonei de gestiune a discului flexibil, nemodificabilă, de zona programabilă de I/E, BIOS, astfel că orice proiectant de sistem îşi va organiza la fel discul, chiar dacă restul periferiei este diferit. Elaborarea unui CP/M propriu conduce la completarea primelor două piste ale dischetei cu programe de sistem, restul discului fiind disponibil pentru interschimb de programe.

Organizarea memoriei, pentru a permite încărcarea CP/M, este descrisă la § 3.5.3.2. CP/M poate fi generat pentru diferite dimensiuni ale memoriei, între 16 Kocteţi şi 64 Kocteţi. Ca sistem de calcul şi de dezvoltare, permite aceleaşi facilităţi ca şi sistemul de operare ISIS-II, cu avantajul că interschimbabilitatea a permis proliferarea mai rapidă a unor noi programe, cu performanţe îmbunătăţite.

Sînt disponibile pentru programele de aplicaţii adresele de la 100H, pînă în zona superioară a memoriei, în care este încărcat sistemul de operare.

Rutinele de sistem sînt grupate în 10 operaţii de bază şi 16 operaţii cu discul (vezi şi § 3.5.3.2):

Cod 1, *Read Console:* citire de la consolă; se preia un caracter ASCII de la consola sistemului.

Cod 2, *Write Console;* scrie la consolă: se trimite un caracter ASCII la consola sistemului.

Cod 3, *Read Reader*: citire de la cititor; se preia un caracter ASCII de la lectorul de bandă perforată.

Cod 4, *Write Punch*: perforează; se trimite un caracter ASCII în vederea perforării pe bandă perforată.

Cod 5, *Write List:* imprimare; se tipăreşte un caracter ASCII la perifericul atribuit pentru listare.

Cod 7, *Read I/O Status:* citire a stării I/E; se citeşte locaţia 3 din memorie, care conţine atribuirea logică a periferiei fizice existente.

Cod 8, *Write I/O Status:* înscrie starea I/E; se înscrie locația 3 din memorie.

Cod 9, *Print Buffer:* tipărește șir de caractere; se trimit la consolă caracterele ASCII dintr-un tampon; ultimul caracter va fi $.

Cod 10, *Read Buffer:* citește de la consolă; se preia de la consolă șirul specificat de caractere ASCII.

Cod 11, *Interrogate Console Ready:* citește starea consolei; se examinează dacă se poate comunica (în intrare sau ieșire) cu consola.

Dintre operațiile cu discul menționăm;

Cod 12, *Lift Head:* descărcare cap; discheta se îndepărtează de capul magnetic.

Cod 13, *Initialization:* inițializare; se inițializează BDOS, se selectează discul A și se programează DMA cu adresa memoriei tampon de la 80H.

Cod 14, *Log-in and Select Disk x:* se consideră x disc de lucru ($x = 0$, 1, 2, 3 etc. pentru A, B, C, D etc.).

Cod 15, *Open File:* deschide fișier; se stabilește FCB* pentru fișierul specificat.

Cod 16, *Close File:* închide fișier; se stabilește FCB inițial.

Cod 17, *Search for File:* caută fișierul specificat.

Cod 18, *Search for Next Occurence:* caută pentru următoarea coincidență; se lansează după operația 17 și se obține adresa următorului FCB din zona DIRECTORY.

Cod 19, *Delete File:* șterge fișier; se elimină de pe dischetă fișierul specificat; înregistrările devin disponibile.

Cod 20, *Read Next Record:* citește următorul sector; se preia înregistrarea din fișierul deschis, cu incrementarea NR* din FCB. După depășirea 0FFH, se deschide automat următoarea extensie și NR este trecut în 00H.

Cod 21, *Write Next Record:* scrie următorul sector; la fel ca la precedenta comandă, dar pentru scriere.

Cod 22, *Make File:* creează un fișier; se creează o intrare în DIRECTORY cu numele de fișier specificat; fișierul creat este gol.

Cod 23, *Rename FCB:* schimbă numele fișierului.

Cod 24, *Interrogate Log-in Vector:* testează discurile în stare READY.

Cod 25, *Interrogate Drive Number:* furnizează adresa discului în lucru.

Cod 26, *Set DMA Address:* programează DMA cu adresa următorului transfer.

Cod 27, *Interrogate Allocation:* furnizează adresa vectorului alocat pentru comanda STATUS.

3.5.3. ORGANIZAREA SPAȚIULUI DE MEMORIE AL MICROCALCULATORULUI

3.5.3.1. LOCAȚII ȘI ZONE DE MEMORIE REZERVATE, COMPATIBILE CU SISTEMUL DE OPERARE ISIS-II.

Programul executat imediat după RESET pregătește cîteva locații la zonele inferioară și superioară ale spațiului memoriei de lucru, sintetizînd caracteristicile structurii sistemului și facilitînd legătura MONITOR-ului cu sistemul de operare și programele de aplicație.

---

* vezi § 3.5.3.2.

Zona inferioară are următoarele locații rezervate:

0÷2 : instrucțiune de salt la rutina RESTART (JMP 0FF0FH), pentru salvarea conținutului registrelor și trecerea spre regimul MONITOR.

3 : octet care descrie atribuirea logică a perifericelor fizice, la un moment dat. Modificări în acest octet permit conectarea, pe rînd, a mai multor periferice fizice pe rolul aceleiași funcții logice. Există 4 periferice logice: consolă, perforator, lector și imprimantă. Fiecăruia i se poate atașa o opțiune „fizică" dintr-un set de maximum 4 opțiuni „fizice".

4÷5 : 2 octeți cu adresa zonei superioare, *top*, a memoriei disponibile. Pentru varianta cu 60 Kocteți se plasează 0EFFFH, iar pentru 62 Kocteți, se va găsi 0F7FFH. O rutină specială, executată după RESET, testează starea memoriei RAM și calculează marginea ei superioară.

8÷0AH : instrucțiune de salt la rutina de reîncărcare a sistemului de operare ISIS-II.

20H÷22H : salt la rutina de tratare a întreruperii IT4, lansate de 8271/8272.

28H÷2AH : salt la rutina de tratare a întreruperii IT5, de achitare a execuției operației lansate pe SSDF.

30H÷32H : salt la rutina de tratare a întreruperii IT6, de execuție a unei operații pe SSDF.

În zona inferioară a spațiului de memorie disponibilă se plasează stiva de lucru, urmată de o rutină de salvare a parametrilor programului întrerupt și de un tabel cu instrucțiuni de salt la diferite subprograme care deservesc perifericele opționale adăugate de utilizator, în afara celor standard, deservite de MONITOR.

Sistemul de operare ISIS-II ocupă spațiul 40H÷2FFFH și are memoria tampon între 3000H÷3680H. De la adresa 3680H pot începe programele utilizator, pînă la zona de stivă din partea superioară a memoriei disponibile. În zona 0F000H÷0F7FFH se află rutinele aferente întreruperilor 4÷6, pentru operații cu discul. Zona 0F800H÷0FFFFH este ocupată de programul MONITOR, care începe cu instrucțiunile de salt la rutinele de sistem:

BEGIN, intrare în MONITOR după RESET;

CI, *Console Input,* intrare de la tastatura consolei atribuite, un caracter;

RI, *Reader Input,* intrare de la cititorul atribuit, un caracter;

CO, *Console Output,* ieșire pe display-ul consolei atribuite, un caracter;

PO, *Punch Output,* ieșire pe perforatorul atribuit, un caracter;

LO, *List Output,* ieșire pe perifericul atribuit la imprimare, un caracter;

CSTS, *Console Status Test,* testare a stării consolei atribuite, pentru a prelua/preda următorul caracter;

IOCHK, *I/O Check,* testare a octetului din locația 3 a memoriei, pentru aflarea configurației de periferice „fizice" atribuite celor 4 periferice „logice";

IOSET, *I/O Set,* modificare a configurației de periferice „fizice" atribuite celor „logice", prin poziționarea corespunzătoare în octetul din locația 3 (IOBYTE);

MEMCK, *Memory Check,* test al memoriei pentru aflarea limitei ei superioare, *top,* înscrise în locațiile 4, 5;

IODEF, *I/O Definition,* stabilire a perifericelor „utilizator" în categoria perifericelor „fizice" disponibile.

3.5.3.2. LOCAŢII ŞI ZONE DE MEMORIE REZERVATE, COMPATIBILE CU SISTEMUL DE OPERARE CP/M

Spre deosebire de ISIS-II, care este un sistem de operare legat de programul MONITOR pentru operaţiile de I/E, sistemul CP/M poate fi independent. Pentru microsistemul din exemplu se folosesc totuşi rutinele existente din MONITOR pentru a simplifica scrierea BIOS-ului, zona de I/E modificabilă a CP/M. Locaţiile şi zonele rezervate sînt proprii CP/M cu 60Kocteţi memorie disponibilă.

0÷2: instrucţiune de salt la reîncărcarea CP/M.

3 : octet configuraţie I/E, ca şi la ISIS-II.

4 : octet cu adresa ultimului UDF adresat.

5÷7: instrucţiune de salt la sistemul de gestiune a discului flexibil BDOS.

20H÷22H, 28H÷2AH, 30H÷32H: instrucţiuni de salt la programele pentru tratarea întreruperilor de disc flexibil, ca şi la sistemul de operare ISIS-II.

38H÷3AH: instrucţiunea de salt la programele utilitare ale CP/M — de exemplu revenirea în DDT, dintr-o buclă a unui program testat sub DDT *(Dynamic Debugging Tool* — program utilitar de depanare).

5CH÷7CH: zonă rezervată, folosită opţional pentru blocul de control al fişierului curent FCB, care are următoarea structură (numărătoare zecimală 0÷32):

0 : ET, *Entry Type*, număr de ordine, normal 0;

1÷8: FN, *File Name*, nume fişier, numai cu caractere ASCII;

9÷11: FT, *File Type*, tip fişier, cu caractere ASCII.

12 : FE, *File Extent*, număr extensie, normal 0;

13÷14: neutilizat, normal 0;

15 : RC, *Record Count*, număr de înregistrări al extensiei curente;

16÷31: DM, *Disk Allocation Map*, tabel cu parametrii discului modificaţi de sistemul de operare.

32 : NR, *Next Recording*, următoarea înregistrare.

80H÷0FFH: spaţiu pentru stivă sau memorie tampon.

100H÷0D3FFM: spaţiu pentru programe de aplicaţii.

0D400H÷0D006H: CCP, *Console Command Processor*, program de conducere interactivă prin consola sistemului.

0DC06÷0EA00H: BD0S, *Basic Disk Operating System*, program de gestiune a fişierelor de pe discul flexibil.

0EA00H÷0EFFFM: BIOS, *Base of Input/Output System*, adresa de bază a programului de I/E, construit de utilizator.

0F000H÷0FFFFH, rămîne la fel organizat ca şi la sistemul de operare ISIS-II: programele de întrerupere pentru discul flexibil, 0F000H÷÷0F7FFH şi MONITOR-ul, pînă la 0FFFFH.

Rutinele programabile de utilizator sînt apelate în sistemul de operare CP/M, la începutul secţiunii BIOS, folosind la adaptarea operaţiilor de I/E după necesităţile şi posibilităţile proiectantului. Tabelul are 15 instrucţiuni de salt la 2 rutine de încărcare a sistemului de operare, 6 rutine I/E pentru consolă, lector, perforator şi imprimantă şi 7 rutine specifice discului.

BOOT şi WBOOT încarcă sistemul de operare şi efectuează iniţializările necesare după RESET (BOOT — *Bootstrap* — încărcare) sau la dorinţa uti-

lizatorului, în timpul funcționării normale (WBOOT — *Warm Bootstrap* — încărcare la "cald").

CONST, CONIN, CONOUT, LIST, PUNCH, READER operează la nivel de octet pentru transferul unui caracter ASCII, cu perifericul corespunzător. Sistemul lucrează cu perifericele „fizice" prin intermediul atribuirii „logice" codificate în octetul IOBYTE, locația 0003.

HOME, SELDSK, SETTRK, SETSEC, SETDMA, READ și WRITE corespund operațiilor cu discul magnetic.

HOME: retragerea pînă la pista 0 a carului discului selectat.

SELDSK: memorează selectarea discului specificat.

SETTRK: memorează pista cu care se va face transferul.

SETSEC: memorează numărul de ordine al sectorului ce se transferă (CP/M operează numai cu cîte un singur sector la o singură operație cu discul, ca și ISIS-II, de altfel).

SETDMA: programează DMA cu parametrii transferului.

READ: se transferă sectorul specificat de pe disc, în memoria principală, la adresa înscrisă în DMA. Se semnalează eventuala eroare.

WRITE: se transferă din memoria principală 80H octeți, începînd de la adresa programată în DMA, pe disc, în sectorul specificat. Se semnalează eventuala eroare.

BIBLIOGRAFIE

1. * * * Mostek, *SDB-80E. Software Development Board. Operations Manual.*
2. * * * Philips, *Data Handbook, Integrated Circuits*, Part 9, March 1982.
3. * * * Zilog, 1982/1983 *Data Book.*
4. * * * Mostek, 1979 *Microcomputer Data Book.*
5. * * * Motorola, MCM 4116, 1978.
6. Winfield, J., Z80 *Interfacing Techniques for Dynamic RAM (Application Note)*, în [4], p. 309–319.
7. * * * Microelectronica, *MADS, Sistem de dezvoltare cu aplicații multiple, M80 UC*, Manual de utilizare hard, 1986.
8. * * * Intel, *Component Data Catalog*, 1980.
9. * * * Intel, *Microprocessor and Peripherals Handbook*, 1983
10. * * * CDC, *Product Specification for Flexible Disk Drive, Model 9404.*
11. * * * Hungarian Optical Works (MOM) — Budapest, *Floppy Disk Storage Mechanism with Double Recording Density, Type MF 6400, EC-5082.*
12. SPĂTARU, A., *Teoria transmisiunii informației*, Editura Didactică și Pedagogică, București, 1983.
13. * * * *HP 9885M/S Flexible Disc Drive Service Manual*, Hewlett-Packard, Fort Collins Division, September 1977.
14. * * * *Diskette Operating System. Microcomputer Development System. MDS-DOS Hardware Reference Manual*, Intel Corp., 1975.
15. * * * *CP/M FDOS & Utilities Manual*, Digital Research, 1978.
16. * * * *ISIS-II User's Guide*, Intel Corp.
17. * * * *8080/8085 Assembly Language Programming Manual*, Intel Corp., 1979.
18. STĂNCESCU, ȘT.; GROSU, V.; RĂDUȚOI, D., *Procedeu și dispozitiv pentru încărcarea de pe disc flexibil a unui sistem de operare*, Brevet de invenție RSR nr. 83527 din 1983.
19. * * * *The TTL Data Book for Design Engineers*, Second Edition, Texas Instruments.

PROGRAMELE SSDF CU LSI 8271

```
CP/M MACRO ASSEM 2.0     #001     MODUL SSDF CU LSI 8271

                         TITLE    'MODUL SSDF CU LSI 8271'
                         PAGE     68

                ;BIT INITIALIZARE 8271

0001 =                   RAZBIT  EQU  1

                ;CODURI COMENZI IN RC

0000 =                   SCAN     EQU     00H;SCAN CU EGALITATE
0004 =                   SCANEX   EQU     04H;SCAN CU INEGALITATE
000A =                   SCR1     EQU     0AH;SCRIERE DATE (1 SECTOR)
000B =                   SCRD     EQU     0BH;SCRIERE DATE
000E =                   SCRDS1   EQU     0EH;SCRIERE DATE SPECIALE (1 SECTOR)
000F =                   SCRDS    EQU     0FH;SCRIERE DATE SPECIALE
0012 =                   CIT1     EQU     12H;CITIRE DATE (1 SECTOR)
0013 =                   CITD     EQU     13H;CITIRE DATE SECTOARE
0016 =                   CITDS1   EQU     16H;CITIRE SECTOARE NORMALE SI SECTOARE SPECIALE (1 SECTOR)
0017 =                   CITDS    EQU     17H;CITIRE DATE SECTOARE SI DATE SECTOARE SPECIALE
001B =                   CITBI    EQU     1BH;CITIRE BLOC IDENTIFICARE
001E =                   CITV1    EQU     1EH;VERIFICARE SECTOARE NORMALE SI SECTOARE SPECIALE (1 SECTOR)
001F =                   CITV     EQU     1FH;VERIFICARE DATE SI DATE SPECIALE
0023 =                   FORM     EQU     23H;FORMATARE PISTA
0029 =                   CAUTP    EQU     29H;CAUTARE PISTA
002C =                   CITUDF   EQU     2CH;CITIRE STARE UDF
0035 =                   SPCF     EQU     35H;SPECIFICA PARAMETRI UDF
003D =                   CITRS    EQU     3DH;CITIRE REGISTRU SPECIAL
003A =                   SCRRS    EQU     3AH;SCRIERE REGISTRU SPECIAL

                ;SUBORDINEA DE SPECIFICARE
000D =                   SPCFIN   EQU 0DH    ; INITIALIZARE
0010 =                   SPPD0    EQU 10H    ; SPECIFICARE PISTE DEFECTE DR0
0018 =                   SPPD1    EQU 18H    ; SPECIFICARE PISTE DEFECTE DR1
00FF =                   DEFPST   EQU 0FFH   ; SPECIFICARE PISTA DEFECTA 0FFH
0004 =                   STEPRT   EQU 4D     ; INTERVALUL INTRE IMPULSURILE DE STEP = 4MS = 4 * 1MS
0014 =                   HSTIME   EQU 20D    ; TIMP DE AMORTIZARE LA ACCESUL PE PISTA = 20MS = 20 * 1MS
0006 =                   NROTAT   EQU 6      ; NUMAR DE ROTATII CU CAPUL INCARCAT DUPA ULTIMA COMANDA
000F =                   HLTIME   EQU 15D    ; TIMP DE AMORTIZARE OSC LA INCARCARE CAP = 60MS : 15 * 4MS
001B =                   GAP3F6   EQU 33D-6  ; GAP 3
0028 =                   GAP5F6   EQU 46D-6  ; GAP 5
001A =                   GAP1F6   EQU 32D-6  ; GAP 1
0000 =                   LUNG     EQU 0      ; LUNGIME SECTOR = 128 OCTETI
001A =                   NRSCPST  EQU 26D    ; NUMAR DE SECTOARE PE O PISTA

                ;LOCATII VECTORI IT IN PAGINA 0
0000 =                   FALSE    EQU     0
FFFF =                   TRUE     EQU     NOT     FALSE

                ;ADRESE REGISTRE

0000 =                   BASIT    EQU     0000H
0020 =                   AVIT4    EQU     BASIT+4*8
0028 =                   AVIT5    EQU     BASIT+5*8
0030 =                   AVIT6    EQU     BASIT+6*8

00F0 =                   RCOM     EQU     0F0H
00F0 =                   RS       EQU     RCOM
00F0 =                   RCS      EQU     RCOM
00F1 =                   RR       EQU     RCS+1
00F1 =                   RPAR     EQU     RCS+1
00F2 =                   RRAZ     EQU     RCS+2

                ;MASTI DE COD COMANDA 8271

003F =                   RCMSCO   EQU     00111111B
```

```
CP/M MACRO ASSEM 2.0      #002     MODUL SSDF CU LSI 8271

                 ;MASCA UDF SELECTAT

0OC0 =                  RCMSEL  EQU     11000000B

                 ;MASCA BITI STARE 8271

0004 =                  RCRQD   EQU 1 SHL 2     ;CERERE DATE
0008 =                  RCRQIT  EQU 1 SHL 3     ;CERERE LT
0010 =                  RSRRFL  EQU 1 SHL 4     ;RR PLIN
0020 =                  RSRPFL  EQU 1 SHL 5     ;RP PLIN
0040 =                  RSRCFL  EQU 1 SHL 6     ;RC PLIN
0080 =                  RSBUSY  EQU 1 SHL 7     ;8271 OCUPAT

                 ;MASCA BITI REZULTAT EXECUTIE

0006 =                  RRCOD   EQU 00000110B   ; COD EROARE
0018 =                  RRTIP   EQU 00011000B   ; TIP EROARE
0020 =                  RRDDS   EQU 00100000B   ; DD

                 ;DESCRIERE TIP

0000 =                  TIPNOR  EQU     0
0008 =                  EREQIT  EQU     1 SHL 3
0010 =                  EREQ80  EQU     2 SHL 3
0018 =                  ERNER   EQU     3 SHL 3

                 ;DESCRIERE COD

                 ;TIPNOR(00)
0000 =                  FINORM  EQU     0
0002 =                  SCMETEQ EQU     1 SHL 1
0004 =                  SCMETNQ EQU     2 SHL 1

                 ;TIPEREQIT(01)
0000 =                  ERCK    EQU     0
0002 =                  ERDMA   EQU     1 SHL 1
0004 =                  ERCRCB  EQU     2 SHL 1
0006 =                  ERCRCS  EQU     3 SHL 1

                 ;TIPEREQ80(10)
0000 =                  ERNRDY  EQU     0
0002 =                  ERWPT   EQU     1 SHL 1
0004 =                  ERTK00  EQU     2 SHL 1
0006 =                  ERWFT   EQU     3 SHL 1
                 ;ADRESA DRIVER PENTRU 8271
0040 =                  DR0     EQU     1 SHL 6
0080 =                  DR1     EQU     1 SHL 7

                 ;DESCRIERE OCTET CITIRE STARE UDF (ORDIN 02CH)

0001 =                  CNTOPI  EQU     1 SHL 0
0002 =                  TRK00   EQU     1 SHL 1
0004 =                  READY0  EQU     1 SHL 2
0008 =                  WRPROT  EQU     1 SHL 3
0010 =                  INDEX   EQU     1 SHL 4
0020 =                  WRFT    EQU     1 SHL 5
0040 =                  READY1  EQU     1 SHL 6

                 ;ADRESE REGISTRE SPECIALE

0006 =                  SECTCR  EQU     006H
0014 =                  HISCAN  EQU     014H
0013 =                  LOSCAN  EQU     013H
0012 =                  PSTCH0  EQU     012H
001A =                  PSTCH1  EQU     01AH
0017 =                  RMOD    EQU     017H
0023 =                  PCUDF   EQU     023H
0022 =                  PSUDF   EQU     022H
```

```
CP/M MACRO ASSEM 2.0      #003     MODUL SSDF CU LSI 8271

0010 =                  PDEF01  EQU     010H
0011 =                  PDEF02  EQU     011H
0018 =                  PDEF11  EQU     018H
0019 =                  PDEF12  EQU     019H

                 ;BITI AI REGISTRULUI DE MOD

00C0 =                  MODSP   EQU     11000000B
0001 =                  MODMA   EQU     00000001B
0002 =                  MODNRH  EQU     00000010B
00C0 =                  DADMA   EQU      MODSP
00C1 =                  NUDMA   EQU     MODMA OR MODSP
00C0 =                  DOUACR  EQU      MODSP
00C2 =                  UNCR    EQU     MODNRH OR MODSP

                 ;BITI REGISTRU PENTRU CONTROL DIRECT UDF

0001 =                  WRENBL  EQU     1 SHL 0
0002 =                  STEP    EQU     1 SHL 1
0004 =                  DIR     EQU     1 SHL 2
0008 =                  HLOAD   EQU     1 SHL 3
0010 =                  LOCURR  EQU     1 SHL 4
0020 =                  FLROPO  EQU     1 SHL 5

                 ;OCTET BASE
0008 =                  CNTPRZ  EQU     1 SHL 3
0004 =                  SFOP    EQU     1 SHL 2
0002 =                  RDYDR1  EQU     1 SHL 1
0001 =                  RDYDR0  EQU     1 SHL 0

0012 =                  ICW1    EQU     12H
0000 =                  ICW2    EQU     0
008E =                  OCW1    EQU     10001110B
00FC =                  AICW2   EQU     0FCH
00FD =                  AICW1   EQU     AICW2+1
00FC =                  AOCW1   EQU     AICW2

                        ;STIVA
F100 =                  ADSTART EQU     0F100H
F0FE =                  BASTK           EQU     ADSTART-2
F0FE =                  NWSTCK          EQU     BASTK
                 ;ADRESE PORT ISIS
0078 =                  BASE            EQU     078H
                 ;ADRESA DMA
00D0 =                  ADRDMA          EQU     0D0H
                 ;LOCATII PORT I/E DMA
0000 =                  VFYDMA   EQU    0
0080 =                  SCRDMA   EQU    1 SHL 7
0040 =                  CITDMA   EQU    1 SHL 6
                 ;LOCATII PORT C/S DMA
0080 =                  AUTOLOAD EQU    1 SHL 7
0040 =                  TCSTOP   EQU    1 SHL 6
0020 =                  EXTWRIT  EQU    1 SHL 5
0010 =                  ROTPRIOR EQU    1 SHL 4
                        IRPC    X,3210
                        ENACH&X EQU     1 SHL X
                        ENDM
0008+=                  ENACH3  EQU     1 SHL 3
0004+=                  ENACH2  EQU     1 SHL 2
0002+=                  ENACH1  EQU     1 SHL 1
0001+=                  ENACH0  EQU     1 SHL 0
F100                    ORG     ADSTART
                 I82716:
                        ;SALVAREA REGISTRELOR
F100 22FEF0             SHLD    NWSTCK
F103 210000             LXI     H,0
```

```
F106 39                 DAD     SP
F107 31FCF0             LXI     SP,NWSTCK-2
F10A E5                 PUSH    H
F10B D5                 PUSH    D
F10C C5                 PUSH    B
F10D F5                 PUSH    PSW
F10E FB                 EI
F10F CD21F1             CALL    EXECUT
F112 F3                 DI
F113 3E20               MVI     A,20H
F115 D3FD               OUT     AICW1
F117 FB                 EI
F118 F1                 POP     PSW
F119 C1                 POP     B
F11A D1                 POP     D
F11B E1                 POP     H
F11C F9                 SPHL
F11D 2AFEF0             LHLD    NWSTCK
F120 C9                 RET
                        ;RUTINA INTRERUPERII "DE EXECUTIE"

               EXECUT:

                        ;BLOCARE
F121 DB78               IN      BASE
F123 F604               ORI     SFOP
F125 D378               OUT     BASE
F127 328EF1             STA     OLDBASE
                        ;AFLA ADRESA BP
F12A DB79               IN      BASE + 1
F12C 6F                 MOV     L,A
F12D DB7A               IN      BASE + 2
F12F 67                 MOV     H,A
F130 2289F1             SHLD    ADIOPB
                        ;COPIAZA BLOCUL DE PARAMETRI
F133 EB                 XCHG
F134 2E0A               MVI     L,10D
F136 0192F1             LXI     B,IOPB
F139 CD80F1             CALL    COPY
                        ;FILTREAZA PRIM SECTOR
F13C 3A96F1             LDA     PRMSCT
F13F E61F               ANI     1FH
F141 3296F1             STA     PRMSCT
                        ;AFLA ADRESA UDF
F144 2193F1             LXI     H,ORDIN
F147 7E                 MOV     A,M
F148 E630               ANI     00110000B
F14A C26DF1             JNZ     ALTUDF
F14D 3E40               MVI     A,DR0
                        ;DIVERSE INITIALIZARI
F14F 328BF1    UDFMEM:  STA     ADRUDF
F152 AF                 XRA     A
F153 3290F1             STA     ERRBYTE
F156 3291F1             STA     ERRCOD
                        ;SALT LA ORDIN
F159 7E                 MOV     A,M
F15A E607               ANI     00000111B
F15C 2172F1             LXI     H,TABORD
F15F 3D        GTBIT:   DCR     A
F160 CA68F1             JZ      GETAD
F163 23                 INX     H
F164 23                 INX     H
F165 C35FF1             JMP     GTBIT
F168 5E        GETAD:   MOV     E,M
F169 23                 INX     H
F16A 56                 MOV     D,M
F16B EB                 XCHG
F16C E9                 PCHL
```



```
F16D 3E80      ALTUDF:  MVI     A,DR1
F16F C34FF1             JMP     UDFMEM
                        ;ADRESELE RUTINELOR
F172 34F2      TABORD:  DW      CAUT
F174 E3F2               DW      FOR
F176 48F2               DW      REC
F178 57F2               DW      CIT
F17A 6FF2               DW      CITVFY
F17C ABF2               DW      SCR
F17E B7F2               DW      SCRDLT
                        ;COPIERE DIN <D> IN <B>
                        ;PE LUNGIME <L>
F180 1A        COPY:    LDAX    D
F181 02                 STAX    B
F182 03                 INX     B
F183 13                 INX     D
F184 2D                 DCR     L
F185 C280F1             JNZ     COPY
F188 C9                 RET
                        ;REZERVARI
F189           ADIOPB:  DS      2
F18B           ADRUDF:  DS      1
F18C           ADRD0:   DS      1
F18D           ADRD1:   DS      1
F18E           OLDBASE: DS      1
F18F           NEWBASE: DS      1
F190           ERRBYTE: DS      1
F191           ERRCOD:  DS      1
F192           IOPB:    DS      1
F193           ORDIN:   DS      1
F194           NRSCT:   DS      1
F195           NRPST:   DS      1
F196           PRMSCT:  DS      1
F197           ADTR:    DS      2
F199           CRTIOP:  DS      1
F19A           ADRNEX:  DS      2
F19C           TABI:    DS      4*26D
                        ;CONTROL BLOC DE PARAMETRI
F204 2194F1    OKIOPB:  LXI     H,NRSCT
F207 7E                 MOV     A,M
F208 47                 MOV     B,A
F209 FE01               CPI     1 ;PRIM SECTOR 1-26
F20B DA23F2             JC      NOKIOP
F20E FE1B               CPI     26D + 1
F210 D223F2             JNC     NOKIOP
F213 23                 INX     H
F214 7E                 MOV     A,M
F215 FE4D               CPI     76D + 1 ;PISTA 0-76
F217 D223F2             JNC     NOKIOP
F21A 23                 INX     H
F21B 7E                 MOV     A,M
F21C 80                 ADD     B
                        ;ULTIM SECTOR 1-26
F21D FE1C               CPI     26D + 2
F21F D223F2             JNC     NOKIOP
F222 C9                 RET
F223 3E08      NOKIOP:  MVI     A,ERADR ;EROARE IN BP
F225 3291F1             STA     ERRCOD
F228 C3ABF4             JMP     FIN
                        ;CONTROL BP IN CAUTARE
F22B 3A95F1    OKIOPC:  LDA     NRPST
F22E FE4D               CPI     76D + 1
F230 D223F2             JNC     NOKIOP
F233 C9                 RET
                        ;CAUTARE
F234 CD2BF2    CAUT:    CALL    OKIOPC
F237 3A8BF1             LDA     ADRUDF
F23A F629               ORI     CAUTP
```

```
CP/M MACRO ASSEM 2.0    #006    MODUL SSDF CU LSI 8271

F23C 57                  MOV    D,A
F23D 3A95F1              LDA    NRPST
F240 5F                  MOV    E,A
F241 CD2DF4              CALL   COMP
F244 76                  HLT
F245 C3ABF4              JMP    FIN
                         ;RECALIBRARE
F248 3A8BF1     REC:     LDA    ADRUDF
F24B F629                ORI    CAUTP
F24D 57                  MOV    D,A
F24E 1E00                MVI    E,0
F250 CD2DF4              CALL   COMP
F253 76                  HLT
F254 C3ABF4              JMP    FIN
                         ;CITIRE
F257 CD04F2     CIT:     CALL   OKIOPB
F25A CD7BF2              CALL   CALCNR
F25D CD8EF2              CALL   SENSCI
                         ;SFIRSIT COMANDA CITIRE
F260 CD96F2     CITFIN:  CALL   PRGDMA
F263 3A8BF1              LDA    ADRUDF
F266 F617                ORI    CITDS
F268 CDCFF2              CALL   PG8271
F26B 76                  HLT
F26C C3ABF4              JMP    FIN
                         ;CITIRE DE VERIFICARE
F26F CD04F2     CITVFY:  CALL   OKIOPB
F272 CD7BF2              CALL   CALCNR
F275 CD95F2              CALL   SENSVF
                         ;SALT LA SFIRSIT COMANDA CITIRE
F278 C360F2              JMP    CITFIN
                ;RUTINA DE CALCUL A TRANSFERURILOR DMA
F27B 3A94F1     CALCNR:  LDA    NRSCT
F27E 018000              LXI    B,80H
F281 210000              LXI    H,0
F284 09         INCA:    DAD    B
F285 3D                  DCR    A
F286 C284F2              JNZ    INCA
F289 2B                  DCX    H
F28A 7C                  MOV    A,H
F28B E63F                ANI    00111111B
F28D C9                  RET
                         ;CITIRE DMA DIN MEMORIE
F28E F640       SENSCI:  ORI    CITDMA
F290 67                  MOV    H,A
F291 C9                  RET
                         ;SCRIERE DMA IN MEMORIE
F292 F680       SENSCR:  ORI    SCRDMA
F294 67                  MOV    H,A
F295 C9         SENSVF:  RET
                         ;PROGRAMARE DMA
F296 EB         PRGDMA:  XCHG
F297 2A97F1              LHLD   ADTR
                OUTDMA:
                         ;CANALUL 2,ADRESE
F29A 7D                  MOV    A,L
F29B D3D4                OUT    ADRDMA+2*2
F29D 7C                  MOV    A,H
F29E D3D4                OUT    ADRDMA+2*2
                         ;CANALUL 2,NUMARATOR
F2A0 7B                  MOV    A,E
F2A1 D3D5                OUT    ADRDMA+2*2+1
F2A3 7A                  MOV    A,D
F2A4 D3D5                OUT    ADRDMA+2*2+1
                         ;VALIDARE CANAL 2
F2A6 3E44                MVI    A,TCSTOP OR ENACH2
F2A8 D3D8                OUT    ADRDMA+4*2
F2AA C9                  RET
```

```
CP/M MACRO ASSEM 2.0    #007    MODUL SSDF CU LSI 8271

                         ;SCRIERE
F2AB CDBFF2     SCR:     CALL   PRGSCR
F2AE F60B                ORI    SCRD
                         ;FINAL COMANDA SCRIERE
F2B0 CDCFF2     SCRFIN:  CALL   PG8271
F2B3 76                  HLT
F2B4 C3ABF4              JMP    FIN
                         ;SCRIE SECTOARE SPECIALE
F2B7 CDBFF2     SCRDLT:  CALL   PRGSCR
F2BA F60F                ORI    SCRDS
F2BC C3B0F2              JMP    SCRFIN
                         ;PROGRAMARE A SCRIERII
F2BF CD04F2     PRGSCR:  CALL   OKIOPB
                         ;CALCUL NUMAR TRANSFERURI DMA
F2C2 CD7BF2              CALL   CALCNR
                         ;STABILESTE SENS SCRIERE
F2C5 CD92F2              CALL   SENSCR
                         ;PROGRAMEAZA DMA
F2C8 CD96F2              CALL   PRGDMA
F2CB 3A8BF1              LDA    ADRUDF
F2CE C9                  RET
                         ;TRIMITE PARAMETRII IN 8271
                         ;TRIMITE IN 8271 PARAMETRII
F2CF 57         PG8271:  MOV    D,A
F2D0 3A95F1              LDA    NRPST
F2D3 5F                  MOV    E,A
F2D4 3A96F1              LDA    PRMSCT
F2D7 47                  MOV    B,A
F2D8 CD1FF4              CALL   COMPP
F2DB 3A94F1              LDA    NRSCT
F2DE 47                  MOV    B,A
F2DF CD26F4              CALL   BP
F2E2 C9                  RET
                         ;FORMATARE
F2E3 CD2BF2     FOR:     CALL   OKIOPC
                         ;STABILESTE ZONA DE RAM CU BI
F2E6 CD14F3              CALL   BIFILD
                         ;PROGRAMEAZA DMA PENTRU TABELA
F2E9 219CF1              LXI    H,TABI
F2EC 116780              LXI    D,8000H+4*26D-1
F2EF CD9AF2              CALL   OUTDMA
                         ;STABILESTE SI TRIMITE PARAMETRII
F2F2 3A8BF1              LDA    ADRUDF
F2F5 F623                ORI    FORM
F2F7 57                  MOV    D,A
F2F8 3A95F1              LDA    NRPST
F2FB 5F                  MOV    E,A
F2FC 061B                MVI    B,GAP3F6
F2FE CD1FF4              CALL   COMPP
0000 =                   LUNGA  EQU     LUNG SHL 5
F301 061A                MVI    B,(LUNGA) OR NRSCPST
F303 CD26F4              CALL   BP
F306 0628                MVI    B,GAP5F6
F308 CD26F4              CALL   BP
F30B 061A                MVI    B,GAP1F6
F30D CD26F4              CALL   BP
F310 76                  HLT
F311 C3ABF4              JMP    FIN
                         ;UMPLE ZONA DE RAM CU [illegible]
F314 219CF1     BIFILD:  LXI    H,TABI
F317 3A95F1              LDA    NRPST
F31A 47                  MOV    B,A
F31B 0E00                MVI    C,0
F31D 110001              LXI    D,0100H
F320 70         FILLBI:  MOV    M,B
F321 23                  INX    H
F322 71                  MOV    M,C
F323 23                  INX    H
```

CP/M MACRO ASSEM 2.0 #008 MODUL SSDF CU LSI 8271

```
F324 72              MOV    M,D
F325 23              INX    H
F326 73              MOV    M,E
F327 23              INX    H
F328 14              INR    D
F329 7A              MOV    A,D
F32A FE1B            CPI    1BH
F32C C220F3          JNZ    FILLBI
F32F 3A92F1          LDA    10PB
F332 E640            ANI    40H
F334 C8              RZ
F335 2A97F1          LHLD   ADTR
F338 EB              XCHG
F339 2E1A            MVI    L,1AH
F33B 019EF1          LXI    B,TABI+2
              ;FORMATARE ALEATUARE. SE MODIFICA
              ;ORDINEA SECTOARELOR CONFORM TABELEI
F33E 1A     LOOPAL:  LDAX   D
F33F 02              STAX   B
F340 13              INX    D
F341 13              INX    D
                     REPT   4
                     INX    B
                     ENDM
F342+03              INX    B
F343+03              INX    B
F344+03              INX    B
F345+03              INX    B
F346 2D              DCR    L
F347 C23EF3          JNZ    LOOPAL
F34A C9              RET

              ;INITIALIZAREA 8271
              ;RESET
F34B CDB7F3 I8271:   CALL   RESET
              ;SESIZARE PREZENTA CONTROLOR 8271
F34E 06C0            MVI    B,DADMA OR DOUACR
F350 CD0AF4          CALL   SCRRSMD
F353 CD12F4          CALL   CITRSMD
F356 FEC0            CPI    DADMA OR DOUACR
F358 CA5FF3          JZ     CONPRZ
              ;NU ESTE PREZENT,SEMNALIZEAZA SO
F35B AF              XRA    A
F35C D378            OUT    BASE
F35E C9              RET
              ;8271 PREZENT,CONTINUA
F35F 3E08   CONPRZ:  MVI    A,CNTPRZ
F361 D378            OUT    BASE
              ;STABILESTE VECTORII DE IT
F363 CD3EF4          CALL   ININIT
              ;SPECIFICA PARAMETRII
F366 CDDBF3          CALL   SPECF
              ;RECALIBRAREA UDF
F369 CDA0F3          CALL   RECINIT
F36C 2E02            MVI    L,2
              ;AFLAREA STARII UDF-URILOR
F36E 166C   UDFST:   MVI    D,DRO OR CITUDF ;INTII DRO
F370 CD87F3          CALL   COMREZ
F373 2D              DCR    L
F374 C26EF3          JNZ    UDFST
F377 E604            ANI    READYO
F379 C493F3          CNZ    DRYO
F37C 16AC            MVI    D,DR1 OR CITUDF ;ACUM DR1
F37E CD87F3          CALL   COMREZ
F381 E640            ANI    READY1
F383 C49BF3          CNZ    DRY1
```

CP/M MACRO ASSEM 2.0 #009 MODUL SSDF CU LSI 8271

```
F386 C9              RET
F387 CDC7F3 COMREZ:  CALL   BSY      ;AFLA REZULTATUL
F38A 7A              MOV    A,D
F38B D3F0            OUT    RCOM
F38D CDC7F3          CALL   BSY
F390 DBF1            IN     RR
F392 C9              RET
F393 0601   DRYO:    MVI    B,RDYDRO      ;DRO READY
F395 DB78   DRY:     IN     BASE
F397 B0              ORA    B
F398 D378            OUT    BASE
F39A C9              RET
F39B 0602   DRY1:    MVI    B,RDYDR1      ;DR1 READY
F39D C395F3          JMP    DRY
              ;RECALIBRARE LA INITIALIZARE
F3A0 1669   RECINIT: MVI    D,DRO OR CAUTP
F3A2 CDABF3          CALL   RECA
F3A5 16A9            MVI    D,DR1 OR CAUTP
F3A7 CDABF3          CALL   RECA
F3AA C9              RET
              ;RECALIBRARE COMANDA
F3AB 1E00   RECA:    MVI    E,0
F3AD CD2DF4          CALL   COMP
F3B0 76              HLT
F3B1 CDC7F3          CALL   BSY
F3B4 DBF1            IN     RR
F3B6 C9              RET
              ;RESETARE SOFT 8271
F3B7 3E01   RESET:   MVI    A,RAZBIT AND TRUE
F3B9 D3F2            OUT    RRAZ
F3BB 3E00            MVI    A,RAZBIT AND FALSE
F3BD D3F2            OUT    RRAZ
F3BF C9              RET
F3C0 DBF0   TEST:    IN     RS;TEST BIT <C> DIN RS
F3C2 A1              ANA    C
F3C3 C2C0F3          JNZ    TEST
F3C6 C9              RET
              ;TEST CONTROLOR OCUPAT
F3C7 0E80   BSY:     MVI    C,RSBUSY
F3C9 C3C0F3          JMP    TEST
              ;TEST RC PLIN
F3CC 0E40   RCFL:    MVI    C,RSRCFL
F3CE C3C0F3          JMP    TEST
              ;TEST RP PLIN
F3D1 0E20   RPFL:    MVI    C,RSRPFL
F3D3 C3C0F3          JMP    TEST
              ;TEST RR PLIN
F3D6 0E10   RRFL:    MVI    C,RSRRFL
F3D8 C3C0F3          JMP    TEST
              ;SPECIFICARE A PARAMETRILOR UDF
F3DB 1635   SPECF:   MVI    D,SPCF
F3DD 1E0D            MVI    E,SPCFIN
F3DF 0604            MVI    B,STEPRT;SRT
F3E1 CD1FF4          CALL   COMPP
F3E4 0614            MVI    B,HSTIME;HST
F3E6 CD26F4          CALL   BP
0060 =               NROTAX EQU     NROTAT SHL 4
F3E9 066F            MVI    B,(NROTAX) OR HLTIME;HLT
F3EB CD026F4         CALL   BP
              ;PISTE DEFECTE UDFO,UDF1
F3EE CDF5F3          CALL   SPCDO
F3F1 CD05F4          CALL   SPCD1
F3F4 C9              RET
F3F5 1E10   SPCDO:   MVI    E,SPPDO
F3F7 1635   SPLBT:   MVI    D,SPCF
F3F9 06FF            MVI    B,DEFPST
```

```
F3FB CD1FF4              CALL   COMPP
F3FE CD26F4              CALL   BP
F401 CD26F4              CALL   BP
F404 C9                  RET
F405 1E18      SPCD1:    MVI    E,SPPD1
F407 C3F7F3              JMP    SPLBT
                         ;SCRIERE IN REGISTRU SPECIAL DE MOD
F40A 167A      SCRRSMD:MVI      D,DRO OR SCRRS
F40C 1E17                MVI    E,RMOD
F40E CD1FF4              CALL   COMPP
F411 C9                  RET
                         ;CITIRE DIN REGISTRU SPECIAL DE MOD
F412 167D      CITRSMD:MVI      D,DRO OR CITRS
F414 1E17                MVI    E,RMOD
F416 CD2DF4              CALL   COMP
F419 CDC7F3              CALL   BSY
F41C DBF1                IN     RR
F41E C9                  RET
                         ;TRIMITE COMANDA SI PARAMETRII
F41F CD2DF4    COMPP:    CALL   COMP
F422 CD26F4              CALL   BP
F425 C9                  RET
F426 CDD1F3    BP:       CALL   RPFL;TRIMITE PARAMETRU
F429 78                  MOV    A,B
F42A D3F1                OUT    RPAR
F42C C9                  RET
                         ;TRIMITE COMANDA SI 1 PARAMETRU
F42D CD37F4    COMP:     CALL   COM
F430 CDD1F3              CALL   RPFL
F433 7B                  MOV    A,E
F434 D3F1                OUT    RPAR
F436 C9                  RET
F437 CDC7F3    COM:      CALL   BSY;TRIMITE COMANDA
F43A 7A                  MOV    A,D
F43B D3F0                OUT    RCOM
F43D C9                  RET

                         ;PROGRAMAREA VECTORILOR DE INTRERUPERE
F43E 3EC3      ININIT:   MVI    A,0C3H
                         IRPC   X,456
                         STA    AVIT$X
                         LXI    H,I8271&X
                         SHLD   AVIT&X+1
                         ENDM
F440+322000              STA    AVIT$4
F443+2169F4              LXI    H,I82714
F446+222100              SHLD   AVIT4+1
F449+322800              STA    AVIT$5
F44C+21E2F4              LXI    H,I82715
F44F+222900              SHLD   AVIT5+1
F452+323000              STA    AVIT$6
F455+2100F1              LXI    H,I82716
F458+223100              SHLD   AVIT6+1
                         IRPC   X,12
                         MVI    A,ICW&X
                         OUT    AICW&X
                         ENDM
F45B+3E12                MVI    A,ICW1
F45D+D3FD                OUT    AICW1
F45F+3E00                MVI    A,ICW2
F461+D3FC                OUT    AICW2
                         ;PROGRAMAREA 8259
F463 3E0E                MVI    A,OCW1
F465 D3FC                OUT    AOCW1
F467 FB                  EI
F468 C9                  RET
                         ;INTRERUPEREA DE ACHITARE
F469 CDC7F3    I82714:   CALL   BSY
```



```
F46C DBF1                IN     RR
F46E 3290F1              STA    ERRBYTE
F471 E6FF                ANI    0FFH
F473 CA8CF4              JZ     FININ
F476 E620                ANI    20H
F478 C293F4              JNZ    ERSNF
F47B 3A90F1              LDA    ERRBYTE
F47E 1F                  RAR
F47F E60F                ANI    0FH
                         ;EXPLOATAREA TABELEI DE ERORI
F481 219BF4              LXI    H,TABERRCOD
F484 5F                  MOV    E,A
F485 1600                MVI    D,0
F487 19                  DAD    D
F488 7E                  MOV    A,M
F489 3291F1              STA    ERRCOD
                         ;ACHITAREA IT
F48C F3        FININ:    DI
F48D 3E20                MVI    A,20H
F48F D3FD                OUT    AICW1
F491 FB                  EI
F492 C9                  RET
F493 3E01      ERSNF:    MVI    A,1
F495 3291F1              STA    ERRCOD
F498 C38CF4              JMP    FININ
               ;TABELA DE CONVERSIE ERORI COD ISIS
               TABERRCOD:
                         REPT   4
                         DB     0       ;OK
                         ENDM
F49B+00                  DB     0       ;OK
F49C+00                  DB     0       ;OK
F49D+00                  DB     0       ;OK
F49E+00                  DB     0       ;OK
F49F 03                  DB     ERSYNC  ;EROARE SINCRO
F4A0 10                  DB     ERITM   ;EROARE RITM
F4A1 0A                  DB     ERCRCBI ;EROARE BI
F4A2 02                  DB     ERCRC   ;EROARE CRC SECT
F4A3 80                  DB     ERNORDY ;NOT READY
F4A4 20                  DB     ERWRPR  ;WR PROTECT
F4A5 04                  DB     ERPOZ   ;TRACK00
F4A6 40                  DB     ERWRIT  ;WR FAULT
                         REPT   4
                         DB     ERFORM  ;SNF
                         ENDM
F4A7+0E                  DB     ERFORM  ;SNF
F4A8+0E                  DB     ERFORM  ;SNF.
F4A9+0E                  DB     ERFORM  ;SNF
F4AA+0E                  DB     ERFORM  ;SNF

                         ;SFIRSIT EXECUTIE
               FIN:
                         ;CITIRE STARE UDF0
F4AB 166C                MVI    D,DRO OR CITUDF
F4AD CD87F3              CALL   COMREZ
F4B0 1F                  RAR
F4B1 1F                  RAR
F4B2 E601                ANI    RDYDRO
F4B4 328FF1              STA    NEWBASE
                         ;CITIRE STARE UDF1
F4B7 16AC                MVI    D,DR1 OR CITUDF
F4B9 CD87F3              CALL   COMREZ
F4BC 1F                  RAR
F4BD 1F                  RAR
F4BE 1F                  RAR
F4BF 1F                  RAR
F4C0 1F                  RAR
F4C1 E602                ANI    RDYDR1
```

```
F4C3 47             MOV     B,A
F4C4 3A8FF1         LDA     NEWBASE
F4C7 B0             ORA     B
F4C8 47             MOV     B,A
F4C9 3A8EF1         LDA     OLDBASE
F4CC E603           ANI     RDYDRO OR RDYDR1
F4CE B8             CMP     B
F4CF CAD9F4         JZ      FINRDYOK
                    ;STABILIRE PORTI ISIS,NOR
F4D2 D37B           OUT     BASE+3
F4D4 3E02           MVI     A,02H
F4D6 D379           OUT     BASE+1
F4D8 C9             RET
                    ;STABILIRE PORTI ISIS,OR
           FINRDYOK:
F4D9 3A91F1         LDA     ERRCOD
F4DC D37B           OUT     BASE+3
F4DE AF             XRA     A
F4DF D379           OUT     BASE+1
F4E1 C9             RET
                    ;COD ERORI ISIS-II
0001 =              ERDLTD  EQU     1 SHL 0
0002 =              ERCRC   EQU     1 SHL 1
0004 =              ERPOZ   EQU     1 SHL 2
0008 =              ERADR   EQU     1 SHL 3
0010 =              ERITM   EQU     1 SHL 4
0020 =              ERWRPR  EQU     1 SHL 5
0040 =              ERWRIT  EQU     1 SHL 6
0080 =              ERNORDY EQU     1 SHL 7
000A =              ERCRCBI EQU     ERADR OR ERCRC
0003 =              ERSYNC  EQU     ERCRC OR ERDLTD
000E =              ERFORM  EQU     ERCRCBI OR ERPOZ
000F =              ERMARK  EQU     ERFORM OR ERDLTD

                                    ;INTRERUPEREA LANSATA DE 8271
F4E2 22FEF0         I83715: SHLD    NWSTCK
F4E5 210000                 LXI     H,0
F4E8 39                     DAD     SP
F4E9 31FCF0                 LXI     SP,NWSTCK-2
F4EC E5                     PUSH    H
F4ED D5                     PUSH    D
F4EE C5                     PUSH    B
F4EF F5                     PUSH    PSW
F4F0 FB                     EI
F4F1 CD03F5                 CALL    STARE
F4F4 F3                     DI
F4F5 3E20                   MVI     A,20H
F4F7 D3FD                   OUT     AICW1
F4F9 FB                     EI
F4FA F1                     POP     PSW
F4FB C1                     POP     B
F4FC D1                     POP     D
F4FD E1                     POP     H
F4FE F9                     SPHL
F4FF 2AFEF0                 LHLD    NWSTCK
F502 C9                     RET
                                    ;DEBLOCARE CONTROLOR
F503 DB78           STARE:  IN      BASE
F505 E6FB                   ANI     NOT SFOP
F507 D378                   OUT     BASE
F509 C9                     RET

F50A                        END
```

PROGRAMELE SSDF CU LSI 8272

```
CP/M MACRO ASSEM 2.0     #001     MODUL SSDF CU LSI 8272

                     TITLE   'MODUL SSDF CU LSI 8272'
                     PAGE    68
               ;MASCA BITI STARE 8272

 0001 =        BSYSEEKO       EQU   1 SHL 0 ;OCUPAT CU CAUTARE PE UDF0
 0002 =        BSYSEEK1       EQU   1 SHL 1 ;OCUPAT CU CAUTERE PE UDF1
 0004 =        BSYSEEK2       EQU   1 SHL 2 ;OCUPAT CU CAUTARE PE UDF2
 0008 =        BSYSEEK3       EQU   1 SHL 3 ;OCUPAT CU CAUTARE PE UDF3
 0010 =        CB             EQU   1 SHL 4 ;SCRIERE SAU CITIRE IN CURS DE EXECUTIE
 0020 =        NDM            EQU   1 SHL 5 ;8272 IN MOD NON DMA
 0040 =        DIO            EQU   1 SHL 6 ;DIRECTIA TRANSFERULUI
                                            ;DIO = "1" DE LA 8272 SPRE MICROPROCESOR
                                            ;DIO = "0" DE LA MICROPROCESOR SPRE 8272
 0080 =        RQM            EQU   1 SHL 7 ;RD PREGATIT DE TRANSFER CU MAGISTRALA
               ;ADRESA DRIVER PENTRU 8272
                     IRPC    X,0123
                     DR&X    EQU     X
                     ENDM
 0000+=              DR0     EQU     0
 0001+=              DR1     EQU     1
 0002+=              DR2     EQU     2
 0003+=              DR3     EQU     3


               ;OCTET BASE
 0008 =              CNTPRZ  EQU      00001000B
 0004 =              SFOP    EQU      00000100B
 0002 =              RDYDR1  EQU      00000010B
 0001 =              RDYDR0  EQU      00000001B

 0012 =              ICW1    EQU      12H
 0000 =              ICW2    EQU      0
 008E =              OCW1    EQU      10001110B
 00FC =              AICW2   EQU      0FCH
 00FD =              AICW1   EQU      AICW2+1
 00FC =              AOCW1   EQU      AICW2

               ;SUBORDINEA DE SPECIFICARE
               ;INTERVALUL DE TIMP INTRE IMPULSURILE DE STEP  -  1MS
 000C =              SRTIME           EQU     0FH-4+1
               ;TIMP DE AMORTIZARE OSC LA DESCARCARE CAP  -  20MS
 0002 =              HUTIME           EQU     20/16+1
               ;TIMP DE AMORTIZARE OSC LA INCARCARE CAP  -  45MS
 0017 =              HLTIME           EQU     45/2+1
               ;LUNGIME        SECTOR
 0000 =              LUNG             EQU     0        ;128 OCTETI
               ;NUMAR DE SECTOARE PE O PISTA
 001A =              NRSCTPST         EQU     26D
 0078 =              BASE             EQU     078H
 00D0 =              ADRDMA           EQU     0D0H
 0080 =              SCRDMA           EQU     1 SHL 7
 0040 =              CITDMA           EQU     1 SHL 6
 0000 =              VFYDMA           EQU     0
 0080 =              AUTOLOAD         EQU     1 SHL 7
 0040 =              TCSTOP           EQU     1 SHL 6
 0020 =              EXTWRIT          EQU     1 SHL 5
 0010 =              ROTPRIOR         EQU     1 SHL 4
                     IRPC             X,0123
                     ENACH&X          EQU     1 SHL X
                     ENDM
 0001+=              ENACH0           EQU     1 SHL 0
 0002+=              ENACH1           EQU     1 SHL 1
 0004+=              ENACH2           EQU     1 SHL 2
 0008+=              ENACH3           EQU     1 SHL 3
 F100                ORG     0F100H
               I82726:
 F100 229DF4         SHLD    SAVHL
 F103 E1             POP     H
```

```
CP/M MACRO ASSEM 2.0    #002    MODUL SSDF CU LSI 8272

F104 CD84F4             CALL    INT$SAVE
F107 CD11F1             CALL    EXECUT
               FININ:
F10A F3                 DI
F10B 3E20               MVI     A,20H
F10D D3FD               OUT     AICW1
F10F FB                 EI
F110 C9                 RET
F111 DB78      EXECUT:  IN      BASE
F113 F604               ORI     SFOP
F115 D378               OUT     BASE
F117 3286F1             STA     OLDBASE
F11A 210000             LXI     H,0
F11D 2274F4             SHLD    RWTBL+1
F120 23                 INX     H
F121 2276F4             SHLD    RWTBL+3
F124 DB79               IN      BASE + 1
F126 6F                 MOV     L,A
F127 DB7A               IN      BASE + 2
F129 67                 MOV     H,A
F12A 2283F1             SHLD    ADIOPB
F12D EB                 XCHG
F12E 2E0A               MVI     L,LUNIOPB
F130 018AF1             LXI     B,IOPB
F133 CD7AF1             CALL    COPY
F136 3A8EF1             LDA     PRMSCT
F139 E61F               ANI     01FH
F13B 328EF1             STA     PRMSCT
F13E 218BF1             LXI     H,ORDIN
F141 7E                 MOV     A,M
F142 E630               ANI     00110000B
F144 C275F1             JNZ     ALTUDF
F147 3E00               MVI     A,DR0
F149 3285F1    UDFMEM:  STA     ADRUDF
F14C 3273F4             STA     DSKNO
F14F AF                 XRA     A
F150 3289F1             STA     ERRCOD
F153 7E                 MOV     A,M
F154 E607               ANI     00000111B
F156 2167F1             LXI     H,TABORD
F159 3D        GTBIT:   DCR     A
F15A CA62F1             JZ      GETAD
F15D 23                 INX     H
F15E 23                 INX     H
F15F C359F1             JMP     GTBIT
F162 5E        GETAD:   MOV     E,M
F163 23                 INX     H
F164 56                 MOV     D,M
F165 EB                 XCHG
F166 E9                 PCHL
F167 3FF2      TABORD:  DW      CAUT
F169 C6F2               DW      FOR
F16B 45F2               DW      REC
F16D 4FF2               DW      CIT
F16F 82F2               DW      CITVFY
F171 89F2               DW      SCR
F173 93F2               DW      SCRDLT
F175 3E01      ALTUDF:  MVI     A,DR1
F177 C349F1             JMP     UDFMEM
F17A 1A        COPY:    LDAX    D
F17B 02                 STAX    B
F17C 03                 INX     B
F17D 13                 INX     D
F17E 2D                 DCR     L
F17F C27AF1             JNZ     COPY
F182 C9                 RET
F183           ADIOPB:  DS      2
F185           ADRUDF:  DS      1
```

```
CP/M MACRO ASSEM 2.0    #003    MODUL SSBF CU LSI 8272

F186           OLDBASE: DS      1
F187           DMALOC:  DS      2
F189           ERRCOD:  DS      1
F18A           IOPB:    DS      1
F18B           ORDIN:   DS      1
F18C           NRSCT:   DS      1
F18D           NRPST:   DS      1
F18E           PRMSCT:  DS      1
F18F           ADIR:    DS      2
F191           CRTIOP:  DS      1
F192           ADRNEX:  DS      2
000A =         LUNIOPB  EQU     $-IOPB
F194           TABI:    DS      4*26D
0068 =         LUNTABI  EQU     $-TABI
F1FC 218CF1    OKIOPB:  LXI     H,NRSCT
F1FF 7E                 MOV     A,M
F200 47                 MOV     B,A
F201 FE01               CPI     1
F203 DA20F2             JC      NOKIOP
F206 FE1B               CPI     26D + 1
F208 D220F2             JNC     NOKIOP
F20B 23                 INX     H
F20C 7E                 MOV     A,M
F20D FE4D               CPI     76D + 1
F20F D220F2             JNC     NOKIOP
F212 23                 INX     H
F213 7E                 MOV     A,M
F214 FE01               CPI     1
F216 DA20F2             JC      NOKIOP
F219 80                 ADD     B
F21A FE1C               CPI     26D + 1 + 1
F21C D220F2             JNC     NOKIOP
F21F C9                 RET
F220 3E08      NOKIOP:  MVI     A,ERADR
F222 3289F1             STA     ERRCOD
F225 C32DF4             JMP     FIN
F228 3A8DF1    OKIOPC:  LDA     NRPST
F22B FE4D               CPI     76D + 1
F22D D220F2             JNC     NOKIOP
F230 C9                 RET
F231 CD28F2    SEEK:    CALL    OKIOPC
F234 3A8DF1    SEEKC:   LDA     NRPST
F237 3274F4             STA     TRKNO
F23A 060F               MVI     B,SKCMD
F23C 0E03               MVI     C,3
F23E C9                 RET
F23F CD31F2    CAUT:    CALL    SEEK
F242 C36FF2             JMP     FINNRW
F245 CD8AF3    REC:     CALL    RECA
F248 AF                 XRA     A
F249 3274F4             STA     TRKNO
F24C C32DF4             JMP     FIN
               CIT:
F24F 3E40               MVI     A,CITDMA
F251 3287F1             STA     DMALOC
F254 0606      CITX:    MVI     B,SRDCMD
F256 C5        CITSCR:  PUSH    B
F257 CDFCF1             CALL    OKIOPB
F25A CD34F2             CALL    SEEKC
F25D CD75F2             CALL    CMDEX
F260 CD98F2             CALL    CALCNR
F263 CDACF2             CALL    PRGDMA
F266 C1                 POP     B
F267 0E00               MVI     C,0
F269 3A8EF1    CMDFIN:  LDA     PRMSCT
F26C 3276F4             STA     SECT
F26F CD75F2    FINNRW:  CALL    CMDEX
F272 C32DF4             JMP     FIN
```

```
F275 CDF3F4    CMDEX:  CALL    CMDRDY
               FL$WAITINT:
               ;VALIDARE IT
F278 FB                EI
F279 76                HLT
F27A 3AA1F4            LDA     INTFL
F27D B7                ORA     A        ;IT VALIDA?
F27E CA78F2            JZ      FL$WAITINT;NU, ASTEAPTA
F281 C9                RET

F282 AF        CITVFY: XRA     A
F283 3287F1            STA     DMALOC
F286 C354F2            JMP     CITX
F289 0605      SCR:    MVI     B,SWRCMD
F28B 3E80      SCRX:   MVI     A,SCRDMA
F28D 3287F1            STA     DMALOC
F290 C356F2            JMP     CITSCR
F293 0609      SCRDLT: MVI     B,SWRSGMD
F295 C38BF2            JMP     SCRX
F298 3A8CF1    CALCNR: LDA     NRSCT
F29B 018000            LXI     B,80H
F29E 210000            LXI     H,0
F2A1 09        INCA:   DAD     B
F2A2 3D                DCR     A
F2A3 C2A1F2            JNZ     INCA
F2A6 2B                DCX     H
F2A7 7C                MOV     A,H
F2A8 E63F              ANI     00111111B
F2AA 67                MOV     H,A
F2AB C9                RET
F2AC 3A87F1    PRGDMA: LDA     DMALOC
F2AF B4                ORA     H
F2B0 67                MOV     H,A
F2B1 EB                XCHG
F2B2 2A8FF1            LHLD    ADTR
F2B5 7D        OUTDMA: MOV     A,L
F2B6 D3D4              OUT     ADRDMA+2*2
F2B8 7C                MOV     A,H
F2B9 D3D4              OUT     ADRDMA+2*2
F2BB 7B                MOV     A,E
F2BC D3D5              OUT     ADRDMA+2*2+1
F2BE 7A                MOV     A,D
F2BF D3D5              OUT     ADRDMA+2*2+1
F2C1 3E44              MVI     A,TCSTOP OR ENACH2
F2C3 D3D8              OUT     ADRDMA+4*2
F2C5 C9                RET


F2C6 CD31F2    FOR:    CALL    SEEK
F2C9 CD75F2            CALL    CMDEX
F2CC CDEBF2            CALL    BIFILD
F2CF 2194F1            LXI     H,TABI
F2D2 116780            LXI     D,8000H+LUNTABI-1
F2D5 CDB5F2            CALL    OUTDMA
F2D8 21001A            LXI     H,PGN OR (PGSC SHL 8)
F2DB 2274F4            SHLD    RWTBL+1
F2DE 211BE5            LXI     H,PGPL OR (PGDS SHL 8)
F2E0 =                 ETIDATA EQU     $-1
F2E1 2276F4            SHLD    RWTBL+3
F2E4 060D              MVI     B,SFRCMD
F2E6 0E06              MVI     C,6
F2E8 C36FF2            JMP     FINNRW
0000 =                 PGN     EQU     0
001A =                 PGSC    EQU     26D
001B =                 PGPL    EQU     27D
00E5 =                 PGDS    EQU     0E5H
F2EB 2194F1    BIFILD: LXI     H,TABI
F2EE 3A8DF1            LDA     NRPST
```



```
F2F1 47                MOV     B,A
F2F2 0E00              MVI     C,0
F2F4 110001            LXI     D,0100H
F2F7 70        FILLBI: MOV     M,B
F2F8 23                INX     H
F2F9 71                MOV     M,C
F2FA 23                INX     H
F2FB 72                MOV     M,D
F2FC 23                INX     H
F2FD 73                MOV     M,E
F2FE 23                INX     H
F2FF 14                INR     D
F300 7A                MOV     A,D
F301 FE1B              CPI     1BH
F303 C2F7F2            JNZ     FILLBI
F306 3EE5              MVI     A,PGDS
F308 32E0F2            STA     ETIDATA
F30B 3A8AF1            LDA     IOFB
F30E E640              ANI     40H
F310 C8                RZ
F311 2A8FF1            LHLD    ADTR
F314 23                INX     H
F315 7E                MOV     A,M
F316 32E0F2            STA     ETIDATA
F319 2B                DCX     H
F31A EB                XCHG
F31B 2E1A              MVI     L,1AH
F31D 0196F1            LXI     B,TABI+2
F320 1A        LOOPAL: LDAX    D
F321 02                STAX    B
F322 13                INX     D
F323 13                INX     D
                       REPT    4
                       INX     B
                       ENDM
F324+03                INX     B
F325+03                INX     B
F326+03                INX     B
F327+03                INX     B
F328 2D                DCR     L
F329 C220F3            JNZ     LOOPAL
F32C C9                RET

00C2 =         ALFA    EQU (SRTIME SHL 4) OR HUTIME
2E00 =         BETA    EQU ((HLTIME SHL 1) SHL 8)
               SPECF:
F32D 21C22E            LXI     H,ALFA OR BETA
F330 2273F4            SHLD    RWTBL
F333 0603              MVI     B,SCYCMD
F335 0E03              MVI     C,3
F337 CDF3F4            CALL    CMDSEND
F33A 210000            LXI     H,0
F33D 2273F4            SHLD    RWTBL
F340 C9                RET
               I8272:
F341 DBF0              IN      FDCMSR
F343 17                RAL
F344 DA4BF3            JC      CONPRZ
F347 AF                XRA     A
F348 D378              OUT     BASE
F34A C9                RET
F34B 3E08      CONPRZ: MVI     A,CNTPRZ
F34D D378              OUT     BASE
F34F CD94F3            CALL    ININIT
F352 CD2DF3            CALL    SPECF
F355 AF                XRA     A
F356 CD65F3            CALL    DRIV
F359 C47BF3            CNZ     DRY0
```

```
 F35C 3E01              MVI     A,1
 F35E CD65F3            CALL    DRIV
 F361 C485F3            CNZ     DRY1
 F364 C9                RET
              DRIV:
 F365 3273F4            STA     DSKNO
 F368 CD8AF3            CALL    RECA
              DROV:
 F36B 0604              MVI     B,SDSCMD
 F36D 0E00              MVI     C,0
 F36F CDF3F4            CALL    CMDRDY
 F372 CD15F5            CALL    CMDRES
 F375 3A7DF4            LDA     RWSTBL
 F378 E620              ANI     READY
 F37A C9                RET
 F37B 0601    DRY0:     MVI     B,RDYDR0
 F37D F5      DRY:      PUSH    PSW
 F37E DB78              IN      BASE
 F380 B0                ORA     B
 F381 D378              OUT     BASE
 F383 F1                POP     PSW
 F384 C9                RET
 F385 0602    DRY1:     MVI     B,RDYDR1
 F387 C37DF3            JMP     DRY
              RECA:
 F38A 0607              MVI     B,RECCMD
 F38C 0E02              MVI     C,2
 F38E CDF3F4            CALL    CMDRDY
 F391 FB                EI
 F392 76                HLT
 F393 C9                RET

 0000 =                 BASIT   EQU     0
 F394 3EC3    ININIT:   MVI     A,0C3H
                        IRPC    X,456
              AVIT$X    EQU     BASIT+X*8
                        STA     AVIT$X
                        LXI     H,I8272$X
                        SHLD    AVIT$X+1
                        ENDM
 0020+=       AVIT$4    EQU     BASIT+4*8
 F396+322000            STA     AVIT$4
 F399+218BF3            LXI     H,I8272$4
 F39C+222100            SHLD    AVIT$4+1
 0028+=       AVIT$5    EQU     BASIT+5*8
 F39F+322800            STA     AVIT$5
 F3A2+2163F4            LXI     H,I8272$5
 F3A5+222900            SHLD    AVIT$5+1
 0030+=       AVIT$6    EQU     BASIT+6*8
 F3A8+323000            STA     AVIT$6
 F3AB+2100F1            LXI     H,I8272$6
 F3AE+223100            SHLD    AVIT$6+1
                        IRPC    X,12
                        MVI     A,ICW&X
                        OUT     AICW&X
                        ENDM
 F3B1+3E12              MVI     A,ICW1
 F3B3+D3FB              OUT     AICW1
 F3B5+3E00              MVI     A,ICW2
 F3B7+D3FC              OUT     AICW2
 F3B9 3E8E              MVI     A,OCW1
 F3BB D3FC              OUT     AOCW1
 F3BD FB                EI
 F3BE C9                RET
              I82724:
 F3BF CDC8F4            CALL    MAININT
 F3C2 3A7DF4            LDA     RWSTBL
 F3C5 57                MOV     D,A
```



```
 F3C6 E6C0              ANI     0C0H
 F3C8 FEC0              CPI     0C0H
 F3CA CCD0F3            CZ      SCHBAS
 F3CD C30AF1            JMP     FININ
 F3D0 AF      SCHBAS:   XRA A
 F3D1 CDECF3            CALL    DRAV
 F3D4 C47BF3            CNZ     DRY0
 F3D7 CCF2F3            CZ      DRN0
 F3DA 3E01              MVI     A,1
 F3DC CDECF3            CALL    DRAV
 F3DF C485F3            CNZ     DRY1
 F3E2 CCF7F3            CZ      DRN1
 F3E5 3A85F1            LDA     ADRUDF
 F3E8 3273F4            STA     DSKNO
 F3EB C9                RET
 F3EC 3273F4  DRAV:     STA     DSKNO
 F3EF C36BF3            JMP     DROV
 F3F2 06FE    DRN0:     MVI     B,LOW NOT RDYDR0
 F3F4 C3F9F3            JMP     DRN
 F3F7 06FD    DRN1:     MVI     B,LOW NOT RDYDR1
 F3F9 F5      DRN:      PUSH    PSW
 F3FA DB78              IN      BASE
 F3FC A0                ANA     B
 F3FD D378              OUT     BASE
 F3FF F1                POP     PSW
 F400 C9                RET
 F401 7E      OCTER:    MOV     A,M
 F402 0608              MVI     B,8
 F404 17      XAL:      RAL
 F405 DC0FF4            CC      ORIER
 F408 13                INX     D
 F409 05                DCR     B
 F40A C204F4            JNZ     XAL
 F40D 23                INX     H
 F40E C9                RET
 F40F F5      ORIER:    PUSH    PSW
 F410 1A                LDAX    D
 F411 B1                ORA     C
 F412 4F                MOV     C,A
 F413 F1                POP     PSW
 F414 C9                RET
              TABERIS:
              ;CONVERSIA DE LA ST-0 LA ISIS-II
 F415 0000     DW 0    ;BIT 7,6 - COD ERDARE:
                       ;   0 0 - FARA ERDARE
                       ;   0 1 - TERMINARE ANORMALA
                       ;   1 0 - COMANDA INVALIDA
                       ;   1 1 - TERMINARE ANORMALA
                       ;DIN CAUZA  SCHIMBARII  READY
 F417 00       DB 0    ;TERMINARE A CAUTARII
 F418 40       DB 40H  ;UDF DEFECT
                       ;RECALIBRARE FARA TRACK 00
 F419 80       DB 80H  ;CIT/SCR PE UDF  NOT READY
                       ;SAU ADRESARE FATA 1 INEXISTENTA
 F41A 00       DB 0    ;STARE CAP
 F41B 0000     DW 0    ;COD ADRESA UDF
              ;CONVERSIE ERORI ST-1
 F41D 08000A10          DB      8,0,0AH,10H
 F421 0004200E          DB      0,4,20H,0EH
              ;CONVERSIE ERORI ST-2
 F425 00010204          DB      0,1,2,4
 F429 0000030F          DB      0,0,3,0FH

              FIN:
 F42D 217BF4            LXI     H,RWSTBL
 F430 7E                MOV     A,M
 F431 E6C0              ANI     0C0H
```

```
F433 CA59F4             JZ      FINRDYOK
F436 0E00               MVI     C,0
F438 1115F4             LXI     D,TABERIS
                        REPT 3
                        CALL    OCTER
                        ENDM
F43B+CD01F4             CALL    OCTER
F43E+CD01F4             CALL    OCTER
F441+CD01F4             CALL    OCTER
F444 79                 MOV     A,C
F445 3289F1             STA     ERRCOD
F448 3A86F1             LDA     OLDBASE
F44B 57                 MOV     D,A
F44C DB78               IN      BASE
F44E BA                 CMP     D
F44F CA59F4             JZ      FINRDYOK
F452 D37B               OUT     BASE+3
F454 3E02               MVI     A,2
F456 C360F4             JMP     SCR79
                FINRDYOK:
F459 3A89F1             LDA     ERRCOD
F45C D37B               OUT     BASE+3
F45E 3E00               MVI     A,0
F460 D379       SCR79:  OUT     BASE+1
F462 C9                 RET
                ;COD ERORI ISIS-II
0001 =                  ERDLTD  EQU     1 SHL 0
0002 =                  ERCRC   EQU     1 SHL 1
0004 =                  ERPOZ   EQU     1 SHL 2
0008 =                  ERADR   EQU     1 SHL 3
0010 =                  ERITM   EQU     1 SHL 4
0020 =                  ERWRPR  EQU     1 SHL 5
0040 =                  ERWRIT  EQU     1 SHL 6
0080 =                  ERNORDY EQU     1 SHL 7
000A =                  ERCRCBI EQU ERADR OR ERCRC
0003 =                  ERSYNC  EQU ERCRC OR ERDLTD
000E =                  ERFORM  EQU ERCRCBI OR ERPOZ
000F =                  ERMARK  EQU ERFORM OR ERDLTD

                182725:
F463 229DF4             SHLD    SAVHL
F466 E1                 POP     H
F467 CD84F4             CALL    INT$SAVE
F46A DB78       STARE:  IN      BASE
F46C E6FB               ANI     NOT SFOP
F46E D378               OUT     BASE
F470 C30AF1             JMP     FININ
                ;CONTROLOR DISC SI PORT DMA

00D0 =          DSKB    EQU 0D0H        ;BAZA DMA
00D4 =          CH1DMA  EQU DSKB+4      ;CANAL 1 DMA
00D5 =          CH1TC   EQU DSKB+5      ;CANAL 1 TC
00D4 =          CH2DMA  EQU DSKB+4      ;CANAL 2 DMA
00D5 =          CH2TC   EQU DSKB+5      ;CANAL 2 TC
00D8 =          DMAST   EQU DSKB+8      ;DMA STARE,COMENZI
00F0 =          FDCMSR  EQU 0F0H        ;REGISTRU DE STARI
00F1 =          FDCDDATA  EQU  0F1H     ;REGISTRU DE DATE
                ;
                ;SET INSTRUCTIUNI 8272
                ;
0003 =          SCYCMD  EQU 03H ;PARAMETII PISTA
0005 =          SWRCMD  EQU 05H ;SIMPLA,SCRIE
0006 =          SRDCMD  EQU 06H ;SIMPLA,CITESTE
0045 =          DWRCMD  EQU 45H ;DUBLA,SCRIE
0046 =          DRDCMD  EQU 46H ;DUBLA,CITESTE
00C5 =          D2WCMD  EQU 0C5H;DUBLA,SCRIE 2 FETE
00C6 =          D2RCMD  EQU 0C6H;DUBLA,CITESTE 2 FETE
0004 =          SDSCMD  EQU 04H ;SENSE DRIVE STATUS
```



```
0007 =          RECCMD  EQU 07H ;RECALIBRARE
0008 =          SISCMD  EQU 08H ;SENSE INTERRUPT
000A =          RIDCMD  EQU 0AH ;CITESTE SECTOR BI
000F =          SKCMD   EQU 0FH ;COMANDA CAUTARE
0009 =          SWRSCMD EQU 09H ;SCRIE DELETED DATA
000D =          SFRCMD  EQU 0DH ;FORMATARE
                ST3MEAN:
0080 =          FT       EQU 1 SHL 7    ;FAULT RESET
0040 =          WP       EQU 1 SHL 6    ;WRITE PROTECT
0020 =          READY    EQU 1 SHL 5    ;READY
0010 =          TRK0     EQU 1 SHL 4    ;TRACK 00
0008 =          TWOSIDE  EQU 1 SHL 3    ;TWO SIDES
0004 =          HEADADD  EQU 1 SHL 2    ;SIDE SELECT
0002 =          US1      EQU 1 SHL 1    ;US1 STATUS
0001 =          US0      EQU 1 SHL 0    ;US0 STATUS

                ;TABELA CU PARAMETRI

                RWIBL:
F473 00         DSKNO   DB      0       ;DISC IN LUCRU
F474 00         TRKNO   DB      0       ;PISTA CURENTA
F475 00         HEAD    DB      0       ;ADRESA CAP
F476 01         SECT    DB      1       ;NR SECTOR
F477 00         N       DB      0       ;COD LUNG
F478 1A         EOT     DB      1AH     ;ULTIM SECTOR
F479 07         GPL     DB      7       ;GAP
F47A 80         DTL     DB      128     ;LUNGIME SECTOR
F47B 05         WRCMD   DB      05      ;SCRIERE
F47C 06         RDCMD   DB      06      ;CITIRE


                ;TABELA CU REZULTAT

                RWSIBL:
F47D 00                 DB      0       ;ST-0
F47E 00                 DB      0       ;ST-1
F47F 00                 DB      0       ;ST-2
F480 00                 DB      0       ;C
F481 00                 DB      0       ;H
F482 00                 DB      0       ;R
F483 00                 DB      0       ;N

                ;SALVARE/RESTAURARE

                INT$SAVE:                       ;SALVARE
F484 229FF4             SHLD    SAVRET
F487 E1                 POP     H
F488 229BF4             SHLD    RETCALL+1
F48B F5                 PUSH    PSW
F48C 210000             LXI     H,0
F48F 39                 DAD     SP      ;STIVA VECHE IN HL
F490 31DAF4             LXI     SP,STACK;NOUA STIVA
F493 C5                 PUSH    B
F494 D5                 PUSH    D
F495 E5                 PUSH    H
                ;ADRESA DE REVENIRE IN STIVA
F496 21BAF4             LXI     H,INT$REST
F499 E5                 PUSH    H
F49A C30000     RETCALL JMP     0       ;INTOARCERE
F49D 0000       SAVHL   DW      0       ;MEMOREAZA HL
F49F 0000       SAVRET  DW      0       ;MEMOREAZA RET
F4A1 00         INTFL   DB      0       ;COD IT
F4A2                    DS      24
F4BA =          STACK   EQU     $       ;STIVA IT

                INT$REST:                       ;RESTAURARE
```

```
F4BA E1              POP     H
F4BB D1              POP     D
F4BC C1              POP     B
F4BD F9              SPHL            ;RESTAURARE STIVA VECHE
F4BE F1              POP     PSW
F4BF 2A9FF4          LHLD    SAVRET
F4C2 E5              PUSH    H
F4C3 2A9DF4          LHLD    SAVHL
F4C6 FB              EI
F4C7 C9              RET
                ;
                ;TRATAREA IT LANSATA DE 8272
                ;
                MAININT:
F4C8 DBF0            IN      FDCMSR  ;CITIRE STARE 8272
F4CA E610            ANI     10H     ;OCUPAT?
F4CC C2D6F4          JNZ     RDWRINT ;DA PREIA REZULTAT
F4CF 0608            MVI     B,SISCMD
F4D1 0E00            MVI     C,0
F4D3 CDF3F4          CALL    CMDSEND ;SESIZARE IT
                RDWRINT:
F4D6 CD15F5          CALL    CMDRES  ;PREIA REZULTAT
F4D9 3A7DF4          LDA     RWSTBL
F4DC E6C0            ANI     0C0H    ;MASCA EROARE
F4DE 47              MOV     B,A
F4DF FEC0            CPI     0C0H    ;SCHIMBARE READY
F4E1 3E00            MVI     A,0
F4E3 CAEFF4          JZ      MAININT$EXIT;IGNORA
F4E6 78              MOV     A,B
F4E7 FE80            CPI     80H     ;COMANDA INVALIDA?
F4E9 3E00            MVI     A,0
F4EB CAEFF4          JZ      MAININT$EXIT;IGNORA
F4EE 2F              CMA
                MAININT$EXIT:
F4EF 32A1F4          STA     INTFL   ;VALID COD IT
F4F2 C9              RET;AL RUTINEI DE TRATARE IT
                ;
                ;TRIMITE COMANDA LA 8272
                ;COMANDA IN B
                ;DACA C=0
                ;TRIMITE PARAMETRII SUPLIMENTARI
                ;DIN RWTBL,CITI CERE 8272
                ;DACA C=N,N NENUL,TRANSFERA N OCT
                ;
                CMDRDY:
                CMDSEND:
F4F3 DBF0            IN      FDCMSR  ;PREIA STARE
F4F5 E610            ANI     10H     ;MASCA READY
F4F7 C2F3F4          JNZ     CMDSEND ;ASTEAPTA ELIB
F4FA 2173F4          LXI     H,RWTBL ;INDICATOR TABELA
F4FD F3              DI
F4FE DBF0       CMDOUT: IN   FDCMSR
F500 17              RAL             ;TEST ROM
F501 D2FEF4          JNC     CMDOUT  ;ASTEAPTA ELIB
F504 17              RAL             ;TEST DIO
F505 D8              RC
F506 78              MOV     A,B     ;PARAMETRUL IN A
F507 D3F1            OUT     FDCDDATA;TRIMITE PARAM
F509 46              MOV     B,M;PREIA OCTET URMATOR
F50A 23              INX     H
F50B 79              MOV     A,C
F50C B7              ORA     A       ; C=0
F50D CAFEF4          JZ      CMDOUT  ;INCA UN OCTET
F510 0D              DCR     C
F511 C2FEF4          JNZ     CMDOUT
F514 C9              RET
```



```
                ;CITIRE REZULTAT DIN 8272
                ;
                CMDRES:
F515 217DF4          LXI     H,RWSTBL;INDICATOR TAB
                CMDRES1:
F518 DBF0            IN      FDCMSR  ;PREIA OCTET
F51A 17              RAL             ;TEST RQM
F51B D218F5          JNC     CMDRES1 ;ASTEAPTA ELIB
F51E 17              RAL             ;TEST DIO
F51F D0              RNC             ;SFIRSIT?
F520 DBF1            IN      FDCDDATA
F522 77              MOV     M,A     ;MEMOREAZA OCTET
F523 23              INX     H
F524 C318F5          JMP     CMDRES1 ;ALT OCTET
F527                 END
```

PROGRAMELE SSDF CU CDFU

```
CP/M MACRO ASSEM 2.0    #001    MODUL SSDF CU CDFU

                    TITLE   'MODUL SSDF CU CDFU'
                    PAGE 68
              ;LOCATII COMENZI PCFL
 0001 =             NSELO   EQU     1
 0002 =             VALO    EQU     2
 0004 =             NDIN    EQU     4
 0008 =             NSTOP   EQU     8
 0010 =             NWRIT   EQU     10H
 0020 =             NHLD    EQU     20H
 0040 =             NLWC    EQU     40H
 0080 =             NUNLC   EQU     80H
              ;LOCATII COMENZI PCSFL
 0001 =             NRZDTA  EQU     1
 0002 =             NMFM    EQU     2
 0004 =             NWFR    EQU     4
 0008 =             NDR0    EQU     8
              ;STARI PSFL
 0001 =             RDYO    EQU     1
 0002 =             HLDM    EQU     2
 0004 =             TRKO    EQU     4
 0008 =             INDX    EQU     8
 0010 =             TCT     EQU     10H
 0020 =             CRCZ    EQU     20H
 0040 =             WPR     EQU     40H
 0080 =             HRQ     EQU     80H
              ;INITIALIZARI PORTI SI LOCATII MEMORIE
              ;INITIALIZARE PAGINA DE INTRERUPERI
 0000 =             BASIT   EQU     00000H

              ;INITIALIZARI PORTI
 0082 =             CPGFL   EQU     82H
 00CC =             ZCSFLM  EQU     0CCH
 00FF =             ZCFL    EQU     0FFH
 00CE =             ZCSFL   EQU     0CEH


              ;INTRODUCERE MACRO
                    PUN     MACRO   ABP,BIT,VAL
                    IN      ADP
                    IF      0-      VAL
                    ORI     BIT
                    OUT     ADP
                    ENDIF
                    IF      NOT     0-VAL
                    ANI     0-1-    BIT
                    OUT     ADP
                    ENDIF
                    ENDM
              ;PROGRAM DE INITIALIZARI CONTROLOR
 E700               ORG     0E700H
 E700 3EFF    PZFL:  MVI A,ZCFL
 E702 D3DC          OUT     PCFL
 E704 3ECE          MVI     A,ZCSFL
 E706 D3DE          OUT     PCSFL
 E708 C9            RET
              INIT:
 E709 3E82    PROGC: MVI    A,CPGFL
 E70B D3DF          OUT     PGFL
 E70D CD00E7        CALL    PZFL
 E710 3EFF          MVI     A,0FFH
 E712 321DE8        STA     ADBSD
 E715 CD50E7        CALL    PREGIT
 E718 CD3FE7        CALL    INICO
 E71B CD26E8        CALL    TSTCIR
 E71E CD42E8        CALL    DRV0
 E721 CD5CE8        CALL    DRV1
                    PUN     PCFL,VALO,1
 E724+DBDC          IN      PCFL
```

```
CP/M MACRO ASSEM 2.0    #002    MODUL SSDF CU CDFU

 E726+F602                  ORI     VALO
 E728+D3DC                  OUT     PCFL
 E72A CD3FEA                CALL    RCALIN
 E72D AF                    XRA     A
 E72E 3223E8                STA     NRPIS1
                            PUN     PCFL,VALO,0
 E731+DBDC                  IN      PCFL
 E733+E6FD                  ANI     0-1-    VALO
 E735+D3DC                  OUT     PCFL
 E737 CD3FEA                CALL    RCALIN
 E73A AF                    XRA     A
 E73B 3224E8                STA     NRPIS2
 E73E C9                    RET
 E73F 2E4A          INICO:  MVI     L,NROCTT
 E741 017CE7                LXI     B,ACONST
 E744 11C6E7                LXI     D,FRMDT
 E747 0A            MOV1:   LDAX    B
 E748 12                    STAX    D
 E749 03                    INX     B
 E74A 13                    INX     B
 E74B 2D                    DCR     L
 E74C C247E7                JNZ     MOV1
 E74F C9                    RET
 E750 212800        PREGIT: LXI     H,BASIT+5*8
 E753 36C3                  MVI     M,0C3H
 E755 016DE7                LXI     B,IT5
 E758 23                    INX     H
 E759 71                    MOV     M,C
 E75A 23                    INX     H
 E75B 70                    MOV     M,B
 E75C 213000                LXI     H,BASIT+6*8
 E75F 01E7E8                LXI     B,IT6
 E762 36C3                  MVI     M,0C3H
 E764 23                    INX     H
 E765 71                    MOV     M,C
 E766 23                    INX     H
 E767 70                    MOV     M,B
 E768 3E00                  MVI     A,0
 E76A D3FC                  OUT     0FCH
 E76C C9                    RET
 E76D F5            IT5:    PUSH    PSW
 E76E DB78                  IN      BASE
 E770 E60B                  ANI     0BH
 E772 D378                  OUT     BASE
 E774 F3                    DI
 E775 3E20                  MVI     A,20H
 E777 D3FD                  OUT     0FDH
 E779 FB                    EI
 E77A F1                    POP     PSW
 E77B C9                    RET
 E77C FEFF          ACONST: DB      0FEH,0FFH
 E77E 00                    DB      0
 E77F 000000000             DW      0,0,0
 E785 FE00000000            DB      0FEH,0,0,0,0
 E78A FFFF                  DW      0FFFFH
 E78C C7                    DB      0C7H
 E78D FFFFFFFFFF            DW      0FFFFH,0FFFFH,0FFFFH
 E793 FFFF                  DW      0FFFFH
 E795 C7                    DB      0C7H
 E796 FFFFFFFFFF            DW      0FFFFH,0FFFFH,0FFFFH
 E79C FFFFFFFF              DW      0FFFFH,0FFFFH
                            REPT    11D
                            DB      0
                            ENDM
 E7A0+00                    DB      0
 E7A1+00                    DB      0
 E7A2+00                    DB      0
 E7A3+00                    DB      0
```

```
CP/M MACRO ASSEM 2.0     #003     MODUL SSBF CU CDFU

 E7A4+00                 DB      0
 E7A5+00                 DB      0
 E7A6+00                 DB      0
 E7A7+00                 DB      0
 E7A8+00                 DB      0
 E7A9+00                 DB      0
 E7AA+00                 DB      0
                         REPT    3
                         DB      0A1H
                         ENDM
 E7AB+A1                 DB      0A1H
 E7AC+A1                 DB      0A1H
 E7AD+A1                 DB      0A1H
 E7AE FB                 DB      0FBH
                         REPT    11D
                         DB      0FFH
                         ENDM
 E7AF+FF                 DB      0FFH
 E7B0+FF                 DB      0FFH
 E7B1+FF                 DB      0FFH
 E7B2+FF                 DB      0FFH
 E7B3+FF                 DB      0FFH
 E7B4+FF                 DB      0FFH
 E7B5+FF                 DB      0FFH
 E7B6+FF                 DB      0FFH
 E7B7+FF                 DB      0FFH
 E7B8+FF                 DB      0FFH
 E7B9+FF                 DB      0FFH
                         REPT    3
                         DB      0AH
                         ENDM
 E7BA+0A                 DB      0AH
 E7BB+0A                 DB      0AH
 E7BC+0A                 DB      0AH
 E7BD FF                 DB      0FFH
                         REPT    8
                         DB      04EH
                         ENDM
 E7BE+4E                 DB      04EH
 E7BF+4E                 DB      04EH
 E7C0+4E                 DB      04EH
 E7C1+4E                 DB      04EH
 E7C2+4E                 DB      04EH
 E7C3+4E                 DB      04EH
 E7C4+4E                 DB      04EH
 E7C5+4E                 DB      04EH
 004A =          NROCTT  EQU     $-ACONST
 E7C6            FRMDT:  DS      NROCTT
 E7D6 =          FRMCK   EQU     FRMDT+16D
 E7C9 =          FPRED   EQU     FRMDT+3
 E7CF =          AMRKF   EQU     FPRED+6
 E7D0 =          ANRPF   EQU     AMRKF+1
 E7D2 =          ANRSF   EQU     ANRPF+2
 E7D9 =          FPREK   EQU     FRMCK+3
 E7DF =          AMKCK   EQU     FPREK+6
 E7EA =          FPRD    EQU     FRMDT+24H
 E7F9 =          FCPDD   EQU     FPRD+15D
 E808 =          FINBDD  EQU     FCPDD+15D
                 ;PROGRAM EXECUTIE IOPB DE LA AIOP
 E810            ADRCC:  DS      1
 E811            ADRCO:  DS      1
 E812            ADNRS:  DS      1
 E813            ADNRP:  DS      1
 E814            ADPRS:  DS      1
 E815            ADLAD:  DS      1
 E816            ADHAD:  DS      1
 E817            ADNRO:  DS      1
 E818            ADRPT:  DS      1
```

```
CP/M MACRO ASSEM 2.0     #004     MODUL SSDF CU CDFU

 E819            ADNOL:  DS      1
 E81A            ADNOH:  DS      1
 E81B            ADNXL:  DS      1
 E81C            ADNXH:  DS      1
 E81D            ADBSD:  DS      1
 E81E            ADSVH:  DS      2
 E820            ADSVD:  DS      2
 E822            NRPISU: DS      1
 E823            NRPIS1: DS      1
 E824            NRPIS2: DS      1
 E825            ERBYT:  DS      1
                 ; ADRESE PORTI
 00DF =          PGFL EQU 0DFH
 00DC =          PCFL EQU PGFL-3
 00DD =          PSFL     EQU    PGFL-2
 00DE =          PCSFL    EQU    PGFL-1
 0078 =          BASE     EQU    78H
                 ;TEST PREZENTA CONTROLOR
 E826 CD00E7     TSTCTR:  CALL PZFL
 E829 3E00                MVI    A,0
 E82B D378                OUT    BASE
 E82D DBDC                IN     PCFL
 E82F E6FF                ANI    0-1
 E831 CA3BE8              JZ     NPREZC
 E834 DB78                IN     BASE
 E836 F608                ORI    8
 E838 D378                OUT    BASE
 E83A C9                  RET
 E83B DB78       NPREZC:  IN     BASE
 E83D E6F7                ANI    0-1-8
 E83F D378                OUT    BASE
 E841 C9                  RET
                 DRV0:    PUN    PCFL,VAL0,1
 E842+DBDC                IN     PCFL
 E844+F602                ORI    VAL0
 E846+D3DC                OUT    PCFL
 E848 CD76E8              CALL   SELRDY
 E84B CA55E8              JZ     NDRY0
 E84E DB78                IN     BASE
 E850 F601                ORI    1
 E852 D378                OUT    BASE
 E854 C9                  RET
 E855 DB78       NDRY0:   IN     BASE
 E857 E6FE                ANI    0-1-1
 E859 D378                OUT    BASE
 E85B C9                  RET
                 DRV1:    PUN    PCFL,VAL0,0
 E85C+DBDC                IN     PCFL
 E85E+E6FD                ANI    0-1-   VAL0
 E860+D3DC                OUT    PCFL
 E862 CD76E8              CALL   SELRDY
 E865 CA6FE8              JZ     NDRY1
 E868 DB78                IN     BASE
 E86A F602                ORI    2
 E86C D378                OUT    BASE
 E86E C9                  RET
 E86F DB78       NDRY1:   IN     BASE
 E871 E6FD                ANI    0-1-2
 E873 D378                OUT    BASE
 E875 C9                  RET
                 SELRDY:  PUN    PCFL,NSEL0,0
 E876+DBDC                IN     PCFL
 E878+E6FE                ANI    0-1-   NSEL0
 E87A+D3DC                OUT    PCFL
 E87C DBDB                IN     PSFL
 E87E F5                  PUSH   PSW
                          PUN    PCFL,NSEL0,1
 E87F+DBDC                IN     PCFL
```

```
CP/M MACRO ASSEM 2.0    #005    MODUL SSDF CU CDFU

E881+F601              ORI    NSELO
E883+D3DC              OUT    PCFL
E885 F1                POP    PSW
E886 E601              ANI    RDYO
E888 C9                RET
                ;SELECTARE MOD,DISC,NRPISU NOI
E889 3A10E8    SELECT: LDA ADRCC
E88C E601              ANI    1
E88E C29AE8            JNZ    SELMFM
                       PUN    PCSFL,NMFM,1
E891+DBDE              IN     PCSFL
E893+F602              ORI    NMFM
E895+D3DE              OUT    PCSFL
E897 C3A0E8            JMP    SELT
               SELMFM: PUN    PCSFL,NMFM,0
E89A+DBDE              IN     PCSFL
E89C+E6FD              ANI    0-1-    NMFM
E89E+D3DE              OUT    PCSFL
E8A0 3A11E8    SELT:   LDA    ADRCO
E8A3 E630              ANI    30H
E8A5 CAB7E8            JZ     DRIVO
                       PUN    PCFL,VALO,0
E8A8+DBDC              IN     PCFL
E8AA+E6FD              ANI    0-1-    VALO
E8AC+D3DC              OUT    PCFL
E8AE 3A24E8            LDA    NRPIS2
E8B1 3222E8            STA    NRPISU
E8B4 C3C3E8            JMP    SELX
               DRIVO:  PUN    PCFL,VALO,1
E8B7+DBDC              IN     PCFL
E8B9+F602              ORI    VALO
E8BB+D3DC              OUT    PCFL
E8BD 3A23E8            LDA    NRPIS1
E8C0 3222E8            STA    NRPISU
E8C3 CDCDE8    SELX:   CALL   VERDY
                       PUN    PCFL,NSELO,0
E8C6+DBDC              IN     PCFL
E8C8+E6FE              ANI    0-1-    NSELO
E8CA+D3DC              OUT    PCFL
E8CC C9                RET
E8CD DB78      VERDY:  IN     BASE
E8CF 47                MOV    B,A
E8D0 E608              ANI    8
E8D2 CA9BE9            JZ     ERRDY
E8D5 DBDC              IN     PCFL
E8D7 E602              ANI    2
E8D9 CCE3E8            CZ     DRV1RD
E8DC 78                MOV    A,B
E8DD E601              ANI    1
E8DF CA9BE9            JZ     ERRDY
E8E2 C9                RET
E8E3 78        DRV1RD: MOV    A,B
E8E4 0F                RRC
E8E5 47                MOV    B,A
E8E6 C9                RET
E8E7 221EE8    IT6:    SHLD   ADSVH
E8EA EB                XCHG
E8EB 2220E8            SHLD   ADSVD
E8EE 210000            LXI    H,0
E8F1 39                DAD    SP
E8F2 EB                XCHG
E8F3 21A6F6            LXI    H,STACK
E8F6 F9                SPHL
E8F7 D5                PUSH   D
E8F8 F5                PUSH   PSW
E8F9 C5                PUSH   B
E8FA D5                PUSH   D
E8FB E5                PUSH   H
```

```
CP/M MACRO ASSEM 2.0    #006    MODUL SSDF CU CDFU

E8FC FB                EI
                ; RESETEAZA LOCATIA PI EROARE
E8FD 3E00              MVI    A,0
E8FF 3225E8            STA    ERBYT
                ; ADIOP DIN  BASE+1 SI BASE+2
E902 DB79              IN     BASE+1
E904 6F                MOV    L,A
E905 DB7A              IN     BASE+2
E907 67                MOV    H,A
                ;TRANSLATAREA ADIOP LA ADRCC
                ;INCEPE EXECUTIA IOPB
E908 4D        SCOUT:  MOV C,L
E909 44                MOV    B,H
E90A 2E0A              MVI    L,0AH
E90C 1110E8            LXI    D,ADRCC
E90F 0A                LDAX   B
E910 12                STAX   D
E911 03                INX    B
E912 13                INX    D
E913 2D                DCR    L
E914 C20FE9            JNZ    SCOUT+7
E917 CD26E8            CALL   TSTCTR
E91A CD42E8            CALL   DRVO
E91D CD5CE8            CALL   DRV1
E920 CD89E8            CALL   SELECT
E923 3A13E8            LDA    ADNRP
E926 FE2B              CPI    43D
E928 D433EB            CNC    LCRNT
E92B 3A10E8    IOPBO:  LDA    ADRCC
E92E E601              ANI    1
E930 C25CE9            JNZ    BRNCH
E933 3A11E8    DECOD:  LDA    ADRCO
E936 E60F              ANI    OFH
E938 FE01              CPI    1
E93A CA6AEC            JZ     SEEK
E93D FE02              CPI    2
E93F CA96F0            JZ     FORMT
E942 FE03              CPI    3
E944 CA4BEA            JZ     RECAL
E947 FE04              CPI    4
E949 CA50EB            JZ     LECB
E94C FE05              CPI    5
E94E CA41EB            JZ     LECBFY
E951 FE06              CPI    6
E953 CA89EF            JZ     WRTWD
E956 FE07              CPI    7
E958 CA89EF            JZ     WRTWD
E95B C7                RST    0
E95C 3A11E8    BRNCH:  LDA    ADRCO
E95F E60F              ANI    OFH
E961 FE01              CPI    1
E963 CA6AEC            JZ     SEEK
E966 FE02              CPI    2
E968 CAF8F2            JZ     FORTM
E96B FE03              CPI    3
E96D CA4BEA            JZ     RECAL
E970 FE04              CPI    4
E972 CA50EB            JZ     LECB
E975 FE05              CPI    5
E977 CA41EB            JZ     LECBFY
E97A FE06              CPI    6
E97C CA89EF            JZ     WRTWD
E97F FE07              CPI    7
E981 CA89EF            JZ     WRTWD
E984 C7                RST    0
                ;RUTINE DE SEMNALARE DE ERORI
               ERCRC:    PUN PCSFL,NDRO,1
```

```
E985+DBDE          IN    PCSFL
E987+F608          ORI   NDRQ
E989+D3DE          OUT   PCSFL
E98B 3E02          MVI   A,2
E98D 3225E8        STA   ERBYT
E990 3A10E8        LDA   ADRCC
E993 E601          ANI   01
E995 C2C3E9        JNZ   CORE
E998 C3DBE9        JMP   FINAL
E99B 3E80  ERRDY:  MVI   A,80H
E99D 3225E8        STA   ERBYT
E9A0 C3DBE9        JMP   FINAL

E9A3 3E04  ERPOZ:  MVI   A,4
E9A5 3225E8        STA   ERBYT
E9A8 C3DBE9        JMP   FINAL
E9AB 3E08  ERPRG:  MVI   A,8
E9AD 3225E8        STA   ERBYT
E9B0 C3DBE9        JMP   FINAL
E9B3 3E20  ERWPR:  MVI   A,20H
E9B5 3225E8        STA   ERBYT
E9B8 C3DBE9        JMP   FINAL
E9BB 3E0E  ERSNF:  MVI   A,0EH
E9BD 3225E8        STA   ERBYT
E9C0 C3DBE9        JMP   FINAL
E9C3 DBDE  CORE:   IN    PCSFL
E9C5 E6F0          ANI   0F0H
E9C7 FEC0          CPI   0C0H
E9C9 C2DBE9        JNZ   FINAL
E9CC 3AA9F7        LDA   MEMORY+3
E9CF FEFB          CPI   0FBH
E9D1 C8            RZ
E9D2 3A13E8        LDA   ADRPT
E9D5 FE01          CPI   01
E9D7 C8            RZ
E9D8 C3DBE9        JMP   FINAL
E9DB 3A15E8 FINAL: LDA   ADNRP
E9DE 3222E8        STA   NRPISU
E9E1 3A11E8 FINRY: LDA   ADRCO
E9E4 E630          ANI   30H
E9E6 CAF2E9        JZ    WORKO
E9E9 3A22E8        LDA   NRPISU
E9EC 3224E8        STA   NRPIS2
E9EF C3F8E9        JMP   FINRC
E9F2 3A22E8 WORKO: LDA   NRPISU
E9F5 3223E8        STA   NRPIS1
E9F8 CD91EA FINRC: CALL  DEHLD
                   PUN   PCFL,NSELO,1
E9FB+DBDC          IN    PCFL
E9FD+F601          ORI   NSELO
E9FF+D3DC          OUT   PCFL
EA01 CD3AEB        CALL  HCRNT
           ;INCARCA COD EROARE IN BASE+3
EA04 3A25E8        LDA   ERBYT
EA07 D37B          OUT   BASE+3
           ;INCARCA IN BASE+1 NATURA IT
EA09 3E00          MVI   A,0
EA0B D379          OUT   BASE+1
           ;SEMNALARE SFIRSIT EXECUTIE
EA0D DB78          IN    BASE
EA0F F604          ORI   4
EA11 D378          OUT   BASE
           ;ACHITARE IT
EA13 F3            DI
EA14 3E20          MVI   A,20H
EA16 D3FD          OUT   0FDH
EA18 FB            EI
           ;REFACERE STACK
```



```
EA19 319CF6        LXI   SP,STACK-0AH
EA1C E1            POP   H
EA1D D1            POP   D
EA1E C1            POP   B
EA1F F1            POP   PSW
EA20 E1            POP   H
EA21 F9            SPHL
EA22 2A20E8        LHLD  ADSVD
EA25 EB            XCHG
EA26 2A1EE8        LHLD  ADSVH
EA29 C9            RET
EA2A DBDD  WPROT:  IN    PSFL
EA2C E640          ANI   WPR
EA2E C2B3E9        JNZ   ERWPR
EA31 C9            RET
           ;RUTINE RECALIBRARE SI CAUTARE
           ;RUTINA EXECUTIE RECALIBRARE
EA32 DBDD  RCALI:  IN PSFL
EA34 E604          ANI   TRK0
EA36 C0            RNZ
EA37 0E01          MVI   C,1
EA39 CDC3EA        CALL  OUTST
EA3C C332EA        JMP   RCALI
EA3F 0E64  RCALIN: MVI   C,100D
EA41 CDC3EA        CALL  OUTST
                   PUN   PCFL,NSELO,1
EA44+DBDC          IN    PCFL
EA46+F601          ORI   NSELO
EA48+D3DC          OUT   PCFL
EA4A C9            RET
           ; RECALIBRARE CU REINTOARCERE
           ;IN PROGRAMUL PRINCIPAL
EA4B CD32EA RECAL: CALL  RCALI
EA4E AF            XRA   A
EA4F 3222E8        STA   NRPISU
EA52 C3E1E9        JMP   FINRY
EA55 3A11E8 HLOD:  LDA   ADRCO
EA58 E6F0          ANI   0F0H
EA5A 47            MOV   B,A
EA5B 3A1DE8        LDA   ADBSD
EA5E A8            XRA   B
EA5F C272EA        JNZ   HLLOD
EA62 DBDD          IN    PSFL
EA64 E602          ANI   HLDM
EA66 CA72EA        JZ    HLLOD
                   PUN   PCFL,NHLD,0
EA69+DBDC          IN    PCFL
EA6B+E6DF          ANI   0-1-   NHLD
EA6D+D3DC          OUT   PCFL
EA6F C37BEA        JMP   HLLOD+9
           HLLOD:  PUN   PCFL,NHLD,0
EA72+DBDC          IN    PCFL
EA74+E6DF          ANI   0-1-   NHLD
EA76+D3DC          OUT   PCFL
EA78 CD98EA        CALL  AST
EA7B 3A11E8        LDA   ADRCO
EA7E E6F0          ANI   0F0H
EA80 321DE8        STA   ADBSD
EA83 C9            RET
           REHLD:  PUN   PCFL,NHLD,1
EA84+DBDC          IN    PCFL
EA86+F620          ORI   NHLD
EA88+D3DC          OUT   PCFL
                   PUN   PCFL,NHLD,0
EA8A+DBDC          IN    PCFL
EA8C+E6DF          ANI   0-1-   NHLD
EA8E+D3DC          OUT   PCFL
EA90 C9            RET
```

```
CP/M MACRO ASSEM 2.0    #009    MODUL SSDF CU CDFU

                  DEHLD:  PUN     PCFL,NHLD,1
EA91+DBDC                 IN      PCFL
EA93+F620                 ORI     NHLD
EA95+D3DC                 OUT     PCFL
EA97 C9                   RET
EA98 3E04         AST:    MVI     A,4
EA9A CDA2EA               CALL    WAITN
EA9D 3D                   DCR     A
EA9E C29AEA               JNZ     AST+2
EAA1 C9                   RET
EAA2 163A         WAITN:  MVI     D,3AH
EAA4 CDACEA               CALL    WTAUN
EAA7 15                   DCR     D
EAA8 C2A4EA               JNZ     WAITN+2
EAAB C9                   RET
EAAC 1E14         WTAUN:  MVI     E,14H
EAAE 1D                   DCR     E
EAAF C2AEEA               JNZ     WTAUN+2
EAB2 C9                   RET

EAB3 3A22E8       HDSTP:  LDA     NRPISU
EAB6 47                   MOV     B,A
EAB7 3A13E8               LDA     ADNRP
EABA 90                   SUB     B
EABB C8                   RZ
EABC 4F                   MOV     C,A
EABD D2DFEA               JNC     INTER
EAC0 2F           AFARA:  CMA
EAC1 3C                   INR     A
EAC2 4F                   MOV     C,A
EAC3 CDEAEA       OUTST:  CALL    UNPOT
EAC6 0D                   DCR     C
EAC7 C2C3EA               JNZ     OUTST
EACA CDCEEA               CALL    INTIR
EACD C9                   RET
EACE F5           INTIR:  PUSH    PSW
EACF 3E01                 MVI     A,1
EAD1 C5           AAAAA:  PUSH    B
EAD2 01D007               LXI     B,07D0H
EAD5 CD26EB               CALL    WAIT
EAD8 C1                   POP     B
EAD9 3D                   DCR     A
EADA C2D1EA               JNZ     AAAAA
EADD F1                   POP     PSW
EADE C9                   RET
EADF CDFAEA       INTER:  CALL    UNPIN
EAE2 0D                   DCR     C
EAE3 C2DFEA               JNZ     INTER
EAE6 CDCEEA               CALL    INTIR
EAE9 C9                   RET
                  UNPOT:  PUN     PCFL,NSELO,0
EAEA+DBDC                 IN      PCFL
EAEC+E6FE                 ANI     0-1-    NSELO
EAEE+D3DC                 OUT     PCFL

                          PUN     PCFL,NSTEP,0
EAF0+DBDC                 IN      PCFL
EAF2+E6F7                 ANI     0-1-    NSTEP
EAF4+D3DC                 OUT     PCFL
EAF6 CD10EB               CALL    DPAS
EAF9 C9                   RET
                  UNPIN:  PUN     PCFL,NSELO,0
EAFA+DBDC                 IN      PCFL
EAFC+E6FE                 ANI     0-1-    NSELO
EAFE+D3DC                 OUT     PCFL

                          PUN     PCFL,NDIN,0
EB00+DBDC                 IN      PCFL
```

```
CP/M MACRO ASSEM 2.0    #010    MODUL SSDF CU CDFU

EB02+E6FB                 ANI     0-1-    NDIN
EB04+D3DC                 OUT     PCFL
                          PUN     PCFL,NSTEP,0
EB06+DBDC                 IN      PCFL
EB08+E6F7                 ANI     0-1-    NSTEP
EB0A+D3DC                 OUT     PCFL
EB0C CD10EB               CALL    DPAS
EB0F C9                   RET
EB10 00           DPAS:   NOP
EB11 00                   NOP
EB12 00                   NOP
EB13 00                   NOP
EB14 00                   NOP
EB15 00                   NOP
                          PUN     PCFL,NSTEP,1
EB16+DBDC                 IN      PCFL
EB18+F608                 ORI     NSTEP
EB1A+D3DC                 OUT     PCFL
                          PUN     PCFL,NDIN,1
EB1C+DBDC                 IN      PCFL
EB1E+F604                 ORI     NDIN
EB20+D3DC                 OUT     PCFL
EB22 CDA2EA               CALL    WAITN
EB25 C9                   RET
EB26 CD2EEB       WAIT:   CALL    LATE
EB29 05                   DCR     B
EB2A C226EB               JNZ     WAIT
EB2D C9                   RET
EB2E 0D           LATE:   DCR     C
EB2F C22EEB               JNZ     LATE
EB32 C9                   RET
                  LCRNT:  PUN     PCFL,NLWC,0
EB33+DBDC                 IN      PCFL
EB35+E6BF                 ANI     0-1-    NLWC
EB37+D3DC                 OUT     PCFL
EB39 C9                   RET
                  HCRNT:  PUN     PCFL,NLWC,1
EB3A+DBDC                 IN      PCFL
EB3C+F640                 ORI     NLWC
EB3E+D3DC                 OUT     PCFL
EB40 C9                   RET
                  ;PROGRAM DE CITIRE SI VERIFICARE
EB41 CD56EB       LECBFY:  CALL TSTDBL
EB44 2A04EF               LHLD    ADCMDM
EB47 3E3F                 MVI     A,3FH
EB49 A4                   ANA     H
EB4A 2204EF               SHLD    ADCMDM
EB4D C3F0EB               JMP     RDVFY
EB50 CD56EB       LECB:   CALL    TSTDBL
EB53 C3F0EB               JMP     RDVFY
EB56 3A10E8       TSTDBL: LDA     ADRCC
EB59 E601                 ANI     1
EB5B C2A7EB               JNZ     DUBLAL
EB5E 218040       SIMPLA: LXI     H,4080H
EB61 223AEF               SHLD    ADCMN
EB64 218180               LXI     H,8081H
EB67 2230EF               SHLD    ADNOBS
EB6A 210140               LXI     H,4001H
EB6D 2208EF               SHLD    ADCMB
EB70 210540               LXI     H,4005H
EB73 2206EF               SHLD    ADCMH
EB76 210840               LXI     H,4008H
EB79 220AEF               SHLD    ADSFH
EB7C 21D9E7               LXI     H,FPREK
EB7F 2232EF               SHLD    ADCSP
EB82 210980               LXI     H,8009H
EB85 220CEF               SHLD    ADPRSC
EB88 21C9E7               LXI     H,FPREB
```

```
EB8B 2236EF           SHLD  ADSCP
EB8E 210780           LXI   H,8007H
EB91 2234EF           SHLD  ADNROP
EB94 218000           LXI   H,80H
EB97 2210EF           SHLD  ADNROB
EB9A 21A9F7           LXI   H,MEMORY+3
EB9D 220EEF           SHLD  ADMARK
EBA0 21D7E7           LXI   H,FRMCK+1
EBA3 2238EF           SHLD  ADINB
EBA6 C9               RET
EBA7 210041  DUBLAL:  LXI   H,4100H
EBAA 223AEF           SHLD  ADCMN
EBAD 210181           LXI   H,8101H
EBB0 2230EF           SHLD  ADNOBS
EBB3 210440           LXI   H,4004H
EBB6 2208EF           SHLD  ADCMB
EBB9 210840           LXI   H,4008H
EBBC 2206EF           SHLD  ADCMH
EBBF 211040           LXI   H,4010H
EBC2 2204EF           SHLD  ADSFH
EBC5 21F9E7           LXI   H,FCPDD
EBC8 2232EF           SHLD  ADCSP
EBCB 210F80           LXI   H,800FH
EBCE 220CEF           SHLD  ADPRSC
EBD1 21EAE7           LXI   H,FPRD
EBD4 2236EF           SHLD  ADSCP
EBD7 210F80           LXI   H,800FH
EBDA 2234EF           SHLD  ADNROP

EBDD 210001           LXI   H,100H
EBE0 2210EF           SHLD  ADNROB
EBE3 21A6F7           LXI   H,MEMORY
EBE6 220EEF           SHLD  ADMARK
EBE9 2106E8           LXI   H,FINBDD
EBEC 2238EF           SHLD  ADINB
EBEF C9               RET
EBF0 AF      RDVFY:   XRA   A
EBF1 322CEF           STA   ERBYTE
EBF4 CD83EE           CALL  TESTP
EBF7 CDFEEE           CALL  PRSFL
EBFA 47               MOV   B,A
EBFB 3212EF           STA   ADNRST
EBFE 3A12E8           LDA   ADNRS
EC01 80               ADD   B
EC02 3214EF           STA   ANRPSN
EC05 2A15E8           LHLD  ADLAD
EC08 2216EF           SHLD  ADTRBT
EC0B 3E34             MVI   A,026D*2
EC0D 3218EF           STA   ACRTS
EC10 CDEEEC           CALL  ADERBT
EC13 CD76EC           CALL  SEEKT
EC16 CD42ED  LOOPA:   CALL  LECBBC
EC19 CD22EE           CALL  ER1S
EC1C 2A10EF           LHLD  ADNROB
EC1F EB               XCHG
EC20 2A16EF           LHLD  ADTRBT
EC23 19               DAD   D
EC24 2216EF           SHLD  ADTRBT
EC27 2112EF           LXI   H,ADNRST
EC2A 34               INR   M
EC2B 3A14EF           LDA   ANRPSN
EC2E BE               CMP   M
EC2F CA41EC           JZ    STAFIN
EC32 3E34             MVI   A,026D*2
EC34 3218EF           STA   ACRTS
EC37 AF               XRA   A
EC38 322CEF           STA   ERBYTE
EC3B CD90EC           CALL  CAUTHB
```



```
EC3E C316EC           JMP   LOOPA
EC41 CDEEEC  STAFIN:  CALL  ADERBT
EC44 CDFEEE           CALL  PRSFL
EC47 3212EF  STAFER:  STA   ADNRST
EC4A 2A1CEF           LHLD  ADBYTB
EC4D 3A12EF           LDA   ADNRST
EC50 4F               MOV   C,A
EC51 3A14EF           LDA   ANRPSN
EC54 47               MOV   B,A
EC55 AF               XRA   A
EC56 B6      ORIT:    ORA   M
EC57 322AEF           STA   TEMPOA
EC5A 23               INX   H
EC5B 0C               INR   C
EC5C 79               MOV   A,C
EC5D B8               CMP   B
EC5E 3A2AEF           LDA   TEMPOA
EC61 C256EC           JNZ   ORIT
EC64 3225E8  JFINAL:  STA   ERBYT
EC67 C3DBE9           JMP   FINAL
EC6A CD56EB  SEEK:    CALL  TSTDBL
EC6D CD76EC           CALL  SEEKT
EC70 CDE8ED           CALL  RAZDMA
EC73 C3DBE9           JMP   FINAL
EC76 CDB3EA  SEEKT:   CALL  HDSTP
EC79 CD55EA           CALL  HLOD
EC7C CD90EC  LECTH:   CALL  CAUTHB
EC7F 3A13E8           LDA   ADNRP
EC82 47               MOV   B,A
EC83 3AAAF7           LDA   MEMORY+4
EC86 B8               CMP   B
EC87 C8               RZ
EC88 3E04             MVI   A,04
EC8A 322CEF           STA   ERBYTE
EC8D C3D0EC           JMP   FILER
EC90 3E34    CAUTHB:  MVI   A,26D*2
EC92 3220EF           STA   ANRHEB
EC95 CDFCEC  CAUTHX:  CALL  LECHD
EC98 3A1AEF           LDA   ACRCSH
EC9B E620             ANI   CRCZ
EC9D C2A9EC           JNZ   DCRNH
ECA0 3AA9F7           LDA   MEMORY+4-1
ECA3 FEFE             CPI   0FEH
ECA5 C2A9EC           JNZ   DCRNH
ECA8 C9               RET
ECA9 CDE8ED  DCRNH:   CALL  RAZDMA
ECAC 3A20EF           LDA   ANRHEB
ECAF 3D               DCR   A
ECB0 3220EF           STA   ANRHEB
ECB3 C295EC           JNZ   CAUTHX
ECB6 3A11E8           LDA   ADRCO
ECB9 E607             ANI   7
ECBB FE01             CPI   1
ECBD C2C8EC           JNZ   ERHLEC
ECC0 3E0A             MVI   A,0AH
ECC2 3225E8           STA   ERBYT
ECC5 C3DBE9           JMP   FINAL
ECC8 3E0A    ERHLEC:  MVI   A,0AH
ECCA 322CEF           STA   ERBYTE
ECCD C3D0EC           JMP   FILER
ECD0 2A1CEF  FILER:   LHLD  ADBYTB
ECD3 EB               XCHG
ECD4 2112EF           LXI   H,ADNRST
ECD7 3A14EF  AINRNR:  LDA   ANRPSN
ECDA BE               CMP   M
ECDB CAE9EC           JZ    STRAX
ECDE 3A2CEF           LDA   ERBYTE
ECE1 EB               XCHG
```

```
 EDE2 77                  MOV    M,A
 EDE3 23                  INX    H
 ECE4 EB                  XCHG
 ECE5 34                  INR    M
 ECE6 C3D7EC              JMP    AINRNR
 ECE9 3E01       STRAX:   MVI    A,1
 ECEB C347EC              JMP    STAFER
 ECEE CDFEEE     ADERBT:  CALL   PRSFL
 ECF1 5F                  MOV    E,A
 ECF2 1600                MVI    D,0
 ECF4 213DEF              LXI    H,T7B
 ECF7 19                  DAD    D
 ECF8 221CEF              SHLD   ADBYTB
 ECFB C9                  RET
 ECFC CDE8ED     LECHD:   CALL   RAZDMA
 ECFF CD84EA              CALL   REHLD
 ED02 AF                  XRA    A
 ED03 321EEF              STA    FLY
 ED06 2A0EEF              LHLD   ADMARK
 ED09 EB                  XCHG
 ED0A 2A06EF              LHLD   ADCMH
 ED0D 4D                  MOV    C,L
 ED0E 44                  MOV    B,H
 ED0F CDE2EE              CALL   DMA2
 ED12 2A32EF              LHLD   ADCSP
 ED15 EB                  XCHG
 ED16 2A0CEF              LHLD   ADPRSC
 ED19 4D                  MOV    C,L
 ED1A 44                  MOV    B,H
 ED1B CDD4EE              CALL   DMA0
 ED1E 11AEF7              LXI    D,MEMORY+3
 ED21 2A0AEF              LHLD   ADSFH
 ED24 4D                  MOV    C,L
 ED25 44                  MOV    B,H
 ED26 CDF0EE              CALL   DMA3
 ED29 3EC4                MVI    A,0C4H
 ED2B D3D8                OUT    0D8H
                          PUN    PCSFL,NDRQ,0
 ED2D+DBDE                IN     PCSFL
 ED2F+E6F7                ANI    0-1-     NDRQ
 ED31+D3DE                OUT    PCSFL
 ED33 CDACEE              CALL   TIC1
 ED36 3E0A                MVI    A,0AH
 ED38 3D                  DCR    A
 ED39 C238ED              JNZ    $-1
 ED3C DBDD                IN     PSFL
 ED3E 321AEF              STA    ACRCSH
 ED41 C9                  RET
 ED42 3AACF7     LECBDC:  LDA    MEMORY+4+C
 ED45 2112EF              LXI    H,ADNRST
 ED48 BE                  CMP    M
 ED49 C216EE              JNZ    LECBV
 ED4C 3E01                MVI    A,01
 ED4E 323CEF              STA    ADNSFY
 ED51 CD55ED              CALL   FLYB
 ED54 C9                  RET
 ED55 DBDD       FLYB:    IN     PSFL
 ED57 E610                ANI    TCT
 ED59 CA55ED              JZ     $-4
 ED5C AF                  XRA    A
 ED5D D3D8                OUT    0D8H
 ED5F 3E03                MVI    A,03H
 ED61 3D                  DCR    A
 ED62 C261ED              JNZ    $-1
 ED65 3A3CEF              LDA    ADNSFY
 ED68 FE01                CPI    01
 ED6A CCF8ED              CZ     PARB
 ED6D C4FFED              CNZ    PARFY
```



```
 ED70 2A0EEF              LHLD   ADMARK
 ED73 EB                  XCHG
 ED74 2A08EF              LHLD   ADCMB
 ED77 4D                  MOV    C,L
 ED78 44                  MOV    B,H
 ED79 CDE2EE              CALL   DMA2
 ED7C 2A16EF              LHLD   ADTRDT
 ED7F EB                  XCHG
 ED80 2A04EF              LHLD   ADCMDM
 ED83 44                  MOV    B,H
 ED84 4D                  MOV    C,L
 ED85 CDF0EE              CALL   DMA3
                          PUN    PCSFL,NDRQ,1
 ED88+DBDE                IN     PCSFL
 ED8A+F608                ORI    NDRQ
 ED8C+D3DE                OUT    PCSFL
 ED8E 3EC4                MVI    A,0C4H
 ED90 D3D8                OUT    0D8H
                          PUN    PCSFL,NDRQ,0
 ED92+DBDE                IN     PCSFL
 ED94+E6F7                ANI    0-1-     NDRQ
 ED96+D3DE                OUT    PCSFL
 ED98 0610                MVI    B,10H
 ED9A DBDD       TSTHRQ:  IN     PSFL
 ED9C E680                ANI    HRQ
 ED9E C2ACED              JNZ    TCTSTB
 EDA1 05                  DCR    B
 EDA2 C29AED              JNZ    TSTHRQ
 EDA5 78                  MOV    A,B
 EDA6 322EEF              STA    AMKBNF
 EDA9 C3E2ED              JMP    BLFIN
 EDAC 78         TCTSTB:  MOV    A,B
 EDAD 322EEF              STA    ANKBNF
 EDB0 3E05                MVI    A,5
 EDB2 3D                  DCR    A
 EDB3 C2B2ED              JNZ    $-1
 EDB6 11AEF7              LXI    D,MEMORY+3
 EDB9 2A0AEF              LHLD   ADSFH
 EDBC 4D                  MOV    C,L
 EDBD 44                  MOV    B,H
 EDBE CDF0EE              CALL   DMA3
 EDC1 3AA9F7              LDA    MEMORY+3
 EDC4 D6F8                SUI    0F8H
 EDC6 3222EF              STA    ATDLDT
 EDC9 CAD1ED              JZ     ABLOK
 EDCC 3AA9F7              LDA    MEMORY+3
 EDCF D6FB                SUI    0FBH

 EDD1 3224EF     ABLOK:   STA    ABLNOK
 EDD4 CDACEE              CALL   TIC1
 EDD7 3E0A                MVI    A,0AH
 EDD9 3D                  DCR    A
 EDDA C2D9ED              JNZ    $-1
 EDDD DBDD                IN     PSFL
 EDDF 3220EF              STA    ACRCSB
 EDE2 3E0C       BLFIN:   MVI    A,0CH
 EDE4 3D                  DCR    A
 EDE5 C2E4ED              JNZ    $-1
 EDE8 AF         RAZDMA:  XRA    A
 EDE9 D3D8                OUT    0D8H
                          PUN    PCSFL,NDRQ,1
 EDEB+DBDE                IN     PCSFL
 EDED+F608                ORI    NDRQ
 EDEF+D3DE                OUT    PCSFL
 EDF1 C9                  RET
 EDF2 3E0E       AERSNF:  MVI    A,0EH
 EDF4 322CEF              STA    ERBYTE
 EDF7 C9                  RET
```

```
CP/M MACRO ASSEM 2.0      #015      MODUL SSDF CU CDFU

 EDF8 2A3AEF     PARB:    LHLD    ADCMN
 EDFB 2204EF              SHLD    ADCMDM
 EDFE C9                  RET
 EDFF 2A10EF     PARFY:   LHLD    ADNROB
 EE02 2204EF              SHLD    ADCMDM
 EE05 C9                  RET
 EE06 3E00       BLFLY:   MVI     A,00
 EE08 323CEF              STA     ADNSFY
 EE0B CD55ED              CALL    FLYB
 EE0E 3A18EF              LDA     ACRTS
 EE11 3D                  DCR     A
 EE12 3218EF              STA     ACRTS
 EE15 C9                  RET
 EE16 CD06EE     LECBV:   CALL    BLFLY
 EE19 CAF2ED              JZ      AERSNF
 EE1C CD90EC              CALL    CAUTHB
 EE1F C342ED              JMP     LECRD•
 EE22 3A22EF     ER1S:    LDA     ATDLDT
 EE25 FE00                CPI     0
 EE27 CC56EE              CZ      PRGDLT
 EE2A 3A24EF              LDA     ABLNC•
 EE2D FE00                CPI     0
 EE2F C45FEE              CNZ     PRGMNF
 EE32 3A26EF              LDA     ACRC•B
 EE35 E620                ANI     CRCZ
 EE37 C468EE              CNZ     PRECRC
 EE3A 3A1AEF              LDA     ACRCSH
 EE3D E620                ANI     CRCZ
 EE3F C471EE              CNZ     PRCRCH
 EE42 3A2EEF              LDA     AMKBNF
 EE45 E6FF                ANI     OFFH
 EE47 CC7AEE              CZ      PRBNF
 EE4A 2A1CEF     MOVETB:  LHLD    ADBYTB
 EE4D 3A2CEF              LDA     ERBYTE
 EE50 77                  MOV     M,A
 EE51 23                  INX     H
 EE52 221CEF              SHLD    ADBYTB
 EE55 C9                  RET
 EE56 3A2CEF     PRGDLT:  LDA     ERBYTE
 EE59 F601                ORI     01
 EE5B 322CEF              STA     ERBYTE
 EE5E C9                  RET
 EE5F 3A2CEF     PRGMNF:  LDA     ERBYTE
 EE62 F603                ORI     03H
 EE64 322CEF              STA     ERBYTE
 EE67 C9                  RET
 EE68 3A2CEF     PRECRC:  LDA     ERBYTE
 EE6B F602                ORI     02
 EE6D 322CEF              STA     ERBYTE
 EE70 C9                  RET
 EE71 3A2CEF     PRCRCH:  LDA     ERBYTE
 EE74 F60A                ORI     0AH
 EE76 322CEF              STA     ERBYTE
 EE79 C9                  RET
 EE7A 3A2CEF     PRBNF:   LDA     ERBYTE
 EE7D F60F                ORI     0FH
 EE7F 322CEF              STA     ERBYTE
 EE82 C9                  RET
 EE83 3A13E8     TE•TP:   LDA     ADNRP
 EE86 FE4D                CPI     77D
 EE88 D2BCEE              JNC     ERPG
 EE8B 3A14E8              LDA     ADPRS
 EE8E E61F                ANI     1FH
 EE90 FE00                CPI     0
 EE92 CABCEE              JZ      ERPG
 EE95 FE1B                CPI     27D
 EE97 D2BCEE              JNC     ERPG
 EE9A 47                  MOV     B,A
```

```
CP/M MACRO ASSEM 2.0      #016      MODUL SSDF CU CDFU

 EE9B 3A12E8              LDA     ADNRS
 EE9E 80                  ADD     B
 EE9F FE1C                CPI     28D
 EEA1 D2BCEE              JNC     ERPG
 EEA4 C9                  RET
 EEA5 CDACEE     TC:      CALL    TTCI
 EEA8 CDB4EE              CALL    TTCO
 EEAB C9                  RET
 EEAC DBDD       TTCI:    IN      PSFL
 EEAE E610                ANI     TCT
 EEB0 CAACEE              JZ      $-4
 EEB3 C9                  RET
 EEB4 DBDD       TTCO:    IN      PSFL
 EEB6 E610                ANI     TCT
 EEB8 C2B4EE              JNZ     $-4
 EEBB C9                  RET
 EEBC 3E08       ERPG:    MVI     A,8
 EEBE 322CEF              STA     ERBYTE
 EEC1 3E01       FIXPAR:  MVI     A,1
 EEC3 3212EF              STA     ADNRST
 EEC6 3E1B                MVI     A,1BH
 EEC8 3214EF              STA     ANRPSN
 EECB 213EEF              LXI     H,T7B+1
 EECE 221CEF              SHLD    ADBYTB
 EED1 C3D0EC              JMP     FILER
 EED4 7B         DMA0:    MOV     A,E
 EED5 D3D0                OUT     0D0H
 EED7 7A                  MOV     A,D
 EED8 D3D0                OUT     0D0H
 EEDA 0B                  DCX     B
 EEDB 79                  MOV     A,C
 EEDC D3D1                OUT     0D1H
 EEDE 78                  MOV     A,B
 EEDF D3D1                OUT     0D1H
 EEE1 C9                  RET
 EEE2 7B         DMA2:    MOV     A,E
 EEE3 D3D4                OUT     0D4H
 EEE5 7A                  MOV     A,D
 EEE6 D3D4                OUT     0D4H
 EEE8 0B                  DCX     B
 EEE9 79                  MOV     A,C
 EEEA D3D5                OUT     0D5H
 EEEC 78                  MOV     A,B
 EEED D3D5                OUT     0D5H
 EEEF C9                  RET
 EEF0 7B         DMA3:    MOV     A,E
 EEF1 D3D6                OUT     0D6H
 EEF3 7A                  MOV     A,D
 EEF4 D3D6                OUT     0D6H
 EEF6 0B                  DCX     B
 EEF7 79                  MOV     A,C
 EEF8 D3D7                OUT     0D7H
 EEFA 78                  MOV     A,B
 EEFB D3D7                OUT     0D7H
 EEFD C9                  RET
 EEFE 3A14E8     PRSFL:   LDA     ADPRS
 EF01 E61F                ANI     1FH
 EF03 C9                  RET
 EF04            ADCMDM:  DS      2
 EF06            ADCMH:   DS      2
 EF08            ADCMB:   DS      2
 EF0A            ADSFH:   DS      2
 EF0C            ADPRSC:  DS      2
 EF0E            ADMARK:  DS      2
 EF10            ADNROB:  DS      2
 EF12            ADNRST:  DS      2
 EF14            ANRPSN:  DS      2
 EF16            ADTRDT:  DS      2
```

```
EF18            ACRTS:  DS      2
EF1A            ACRCSH: DS      2
EF1C            ADBYTB: DS      2
EF1E            FLY:    DS      2
EF20            ANRHED: DS      2
EF22            ATDLDT: DS      2
EF24            ABLNOK: DS      2
EF26            ACRCSB: DS      2
EF28            ADTMPB: DS      2
EF2A            TEMPOA: DS      2
EF2C            ERBYTE: DS      2
EF2E            AMKBNF: DS      2
EF30            ADNOBS: DS      2
EF32            ADCSP:  DS      2
EF34            ADNROP: DS      2
EF36            ADSCP:  DS      2
EF38            ADINB:  DS      2
EF3A            ADCMN:  DS      2
EF3C            ADNSFY: DS      1
EF3D            T7B:    DS      1BH
                ;PROGRAM DE SCRIERE

EF58 3E20       EWPR:    MVI A,020H
EF5A 322CEF             STA     ERBYTE
EF5D C3D0EC             JMP     FILER
                SWFR:   PUN     PCFL,NWFR,0
EF60+DBDC               IN      PCFL
EF62+E6FB               ANI     0-1-    NWFR
EF64+D3DC               OUT     PCFL
                        PUN     PCFL,NWFR,
EF66+DBDC               IN      PCFL
EF68+F604               ORI     NWFR
EF6A+D3DC               OUT     PCFL
EF6C C9                 RET
EF6D 3EF8       DELDT:  MVI     A,0F8H
EF6F 320FE7             STA     AMRKF
EF72 C9                 RET
EF73 CDE0EF     TESTW:  CALL    TSTWP
EF76 CD60EF             CALL    SWFR
EF79 3A11E8             LDA     ADRC0
EF7C E607               ANI     07H
EF7E FE07               CPI     07H
EF80 F5                 PUSH    PSW
EF81 CC6DEF             CZ      DELDT
EF84 F1                 POP     PSW
EF85 C476F0             CNZ     NOMDT
EF88 C9                 RET
EF89 CD56EB     WRTWD:  CALL    TSTDBL
EF8C AF                 XRA     A
EF8D 322CEF             STA     ERBYTE
EF90 CD83EE             CALL    TESTP
EF93 CDFEEE             CALL    PRSPL
EF96 47                 MOV     B,A
EF97 3212EF             STA     ADNRST
EF9A 3A12E8             LDA     ADNRS
EF9D 80                 ADD     B
EF9E 3214EF             STA     ANRPSN
EFA1 2A15E8             LHLD    ADLAD
EFA4 2216EF             SHLD    ADTRDT
EFA7 3E34               MVI     A,026D*2
EFA9 3218EF             STA     ACRTS
EFAC CDEEEC             CALL    ADERBT
EFAF CD73EF             CALL    TESTW
EFB2 CD74EC             CALL    SEEKT
EFB5 CD7CF0     LOOPAW: CALL    WRBDC
EFB8 CD4AEE             CALL    MOVETB
EFBB 2A10EF             LHLD    ADNROB
EFBE EB                 XCHG
```



```
EFBF 2A16EF             LHLD    ADTRDT
EFC2 19                 DAD     D
EFC3 2216EF             SHLD    ADTRDT
EFC6 2112EF             LXI     H,ADNRST
EFC9 34                 INR     M
EFCA 3A14EF             LDA     ANRPSN
EFCD BE                 CMP     M
EFCE CA41EC             JZ      STAFIN
EFD1 3E34               MVI     A,026D*2
EFD3 3218EF             STA     ACRTS
EFD6 AF                 XRA     A
EFD7 322CEF             STA     ERBYTE
EFDA CD90EC             CALL    CAUTHB
EFDD C3B5EF             JMP     LOOPAW
EFE0 DBDB       TSTWP:  IN      PSFL
EFE2 E640               ANI     WPR
EFE4 C258EF             JNZ     EWPR
EFE7 C9                 RET
EFE8 AF         SCRUNB: XRA     A
EFE9 D3D8               OUT     0D8H
                        PUN     PCSFL,NDR0,1
EFEB+DBDE               IN      PCSFL
EFED+F608               ORI     NDR0
EFEF+D3DE               OUT     PCSFL
EFF1 2A36EF             LHLD    ADSCP
EFF4 EB                 XCHG
EFF5 2A34EF             LHLD    ADNROP
EFF8 4D                 MOV     C,L
EFF9 44                 MOV     B,H
EFFA CDE2EE             CALL    DMA2
EFFD 3EC5               MVI     A,0C5H
EFFF D3D8               OUT     0D8H
F001 3E05               MVI     A,05H
F003 3D                 DCR     A
F004 C203F0             JNZ     $-1
F007 F3                 DI
                        PUN     PCFL,NWRIT,0
F008+DBDC               IN      PCFL
F00A+E6EF               ANI     0-1-    NWRIT
F00C+D3DC               OUT     PCFL
                        PUN     PCSFL,NDR0,0
F00E+DBDE               IN      PCSFL
F010+E6F7               ANI     0-1-    NDR0
F012+D3DE               OUT     PCSFL
F014 2A16EF             LHLD    ADTRDT
F017 EB                 XCHG
F018 2A30EF             LHLD    ADNOBS
F01B 4D                 MOV     C,L
F01C 44                 MOV     B,H
F01D CDF0EE             CALL    DMA3
F020 CDACEE             CALL    TTC1
F023 3E0A               MVI     A,0AH
F025 3D                 DCR     A
F026 C225F0             JNZ     $-1
F029 2A38EF             LHLD    ADINB
F02C EB                 XCHG
F02D 010880             LXI     B,8008H
F030 CDF0EE             CALL    DMA3
                        PUN     PCSFL,NRZDTA,1
F033+DBDE               IN      PCSFL
F035+F601               ORI     NRZDTA
F037+D3DE               OUT     PCSFL
F039 CD66F0             CALL    IFINW
F03C CDACEE             CALL    TTC1
F03F 79                 MOV     A,C
F040 3D                 DCR     A
F041 C240F0             JNZ     $-1
                        PUN     PCSFL,NRZDTA,0
```

```
 F044+DBDE                IN     PCSFL
 F046+E6FE                ANI    0-1-    NRZDTA
 F048+D3DE                OUT    PCSFL
 F04A 00                  NOP
 F04B 00                  NOP
 F04C 00                  NOP
 F04D 00                  NOP
 F04E 00                  NOP
                          PUN    PCFL,NWRIT,1
 F04F+DBDC                IN     PCFL
 F051+F610                ORI    NWRIT
 F053+D3DC                OUT    PCFL
                          PUN    PCSFL,NDRQ,1
 F055+DBDE                IN     PCSFL
 F057+F608                ORI    NDRQ
 F059+D3DE                OUT    PCSFL
 F05B AF                  XRA    A
 F05C D3D8                OUT    0D8H
 F05E FB                  EI
 F05F 3E20                MVI    A,020H
 F061 3D                  DCR    A
 F062 C261F0              JNZ    $-1
 F065 C9                  RET
 F066 3A10E8    IFINW:    LDA    ADRCC
 F069 E601                ANI    1
 F06B C272F0              JNZ    IFINWD
 F06E 3E0A                MVI    A,0AH
 F070 4F                  MOV    C,A
 F071 C9                  RET
 F072 3E03      IFINWD:   MVI    A,3
 F074 4F                  MOV    C,A
 F075 C9                  RET
 F076 3EFB      NOMDT:    MVI    A,0FBH
 F078 32CFE7              STA    AMRKF
 F07B C9                  RET
 F07C 3AACF7    WRBDC:    LDA    MEMORY+4+2
 F07F 2112FF              LXI    H,ADNRST
 F082 BE                  CMP    M
 F083 C28AF0              JNZ    LECBVW
 F086 CDE8EF              CALL   SCRUNB
 F089 C9                  RET
 F08A CD06EE    LECBVW:   CALL   BLFLY
 F08D CAF2ED              JZ     AERSNF
 F090 CD90EC              CALL   CAUTHB
 F093 C37CF0              JMP    WRBDC
                ;
                ;
                ;
                ;FORMATARE IN DENSITATE SIMPLA
                ;
                ;
                ;
 F096 3A13E8    FORMT:     LDA ADNRP
 F099 FE4D                 CPI    77D
 F09B D2ABE9               JNC    ERPRG
 F09E CDDBF2               CALL   TSTWPF
 F0A1 CD3FE7               CALL   INICO
 F0A4 3A10E8               LDA    ADRCC
 F0A7 E640                 ANI    40H
 F0A9 C2BCF0               JNZ    FALEA
                ;COMPLETARE TABEL FORMATARE NORMALA
 F0AC 213EEF    FNORM:     LXI H,T7B+1
 F0AF 3E01                 MVI    A,1
 F0B1 77        CT7B:      MOV    M,A
 F0B2 23                   INX    H
 F0B3 3C                   INR    A
 F0B4 FE1B                 CPI    1BH
 F0B6 C2B1F0               JNZ    CT7B
```



```
 F0B9 C3DAF0               JMP    FORM
                ; TABEL FORMATARE ALEATOARE LA T7B
 F0BC 2A1CE8    FALEA:     LHLD   ADLAD
 F0BF 23                   INX    H
 F0C0 22F1F2               SHLD   AADEA
 F0C3 2B                   DCX    H
 F0C4 EB                   XCHG
 F0C5 013EEF               LXI    B,T7B+1
 F0C8 CDCEF0               CALL   MOV2
 F0CB C3DAF0               JMP    FORM
 F0CE 2E1A      MOV2:      MVI    L,1AH
 F0D0 1A        REIA:      LDAX   D
 F0D1 02                   STAX   B
 F0D2 13                   INX    D
 F0D3 13                   INX    D
 F0D4 03                   INX    B
 F0D5 2D                   DCR    L
 F0D6 C2D0F0               JNZ    REIA
 F0D9 C9                   RET
                ;FORMATARE PISTA
 F0DA 3A13E8    FORM:      LDA ADNRP
 F0DD FE00                 CPI    0
 F0DF CAEBF0               JZ     ALPHA
 F0E2 CD83EA               CALL   HDSTP
 F0E5 CD55EA    BETA:      CALL   HLOD
 F0E8 C3FAF0               JMP    FOR
 F0EB DBDD      ALPHA:     IN     PSFL
 F0ED E604                 ANI    TRK0
 F0EF C2E5F0               JNZ    BETA
 F0F2 0E01                 MVI    C,1
 F0F4 CDC3EA               CALL   OUTST
 F0F7 C3EBF0               JMP    ALPHA
 F0FA 3A13E8    FOR:       LDA    ADNRP
 F0FD 32D0E7               STA    ANRPF
 F100 213DEF               LXI    H,T7B
 F103 22F6F2               SHLD   LT7B
 F106 3ED7                 MVI    A,0D7H  ;MARK INDEX
 F108 32DFE7               STA    AMKCK
 F10B 3EFC                 MVI    A,0FCH
 F10D 32CFE7               STA    AMRKF
 F110 3ECE                 MVI    A,ZCSFL
 F112 D3DE                 OUT    PCSFL
                ;PREG DATE PRIM BLOC
 F114 CD9CF2               CALL   CD
                ;PREGATIRE DMA INCEPUT PISTA
 F117 11E0E7               LXI    D,AMKCK+1
 F11A 010280               LXI    B,8002H
 F11D CDD4EE               CALL   DMA0
 F120 11E2E7               LXI    D,AMKCK+3
 F123 010280               LXI    B,8002H
 F126 CDE2EE               CALL   DMA2
 F129 3E45                 MVI    A,45H
 F12B D3D8                 OUT    0D8H
 F12D CDCCF2               CALL   TFCIX
                           PUN    PCSFL,NDRQ,0
 F130+DBDE                 IN     PCSFL
 F132+E6F7                 ANI    0-1-    NDRQ
 F134+D3DE                 OUT    PCSFL
                           PUN    PCFL,NWRIT,0
 F136+DBDC                 IN     PCFL
 F138+E6EF                 ANI    0-1-    NWRIT
 F13A+D3DC                 OUT    PCFL
                ;INTIRZIERE SCR. GAP MARK INDEX
 F13C 1E8C                 MVI    E,8CH
 F13E 1D                   DCR    E
 F13F C23EF1               JNZ    $-1
                ;PREGATIRE DMA ZONA MARK INDEX
 F142 11D9E7               LXI    D,FPREK
```

```
F145 010A80          LXI    B,800AH
F148 CDD4EE          CALL   DMA0
F14B 11C9E7          LXI    D,FRMDT+3
F14E 010780          LXI    B,8007H
F151 CDE2EE          CALL   DMA2
F154 11D9E7          LXI    D,FPREK
F157 010380          LXI    B,8003H
F15A CDF0EE          CALL   DMA3
F15D 3EC5            MVI    A,0C5H  ;LANS. DMA
F15F D3D8            OUT    0D8H    ;
               ;INT. DEPASIRE TC CAN2
F161 1E1C            MVI    E,1CH
F163 1D              DCR    E
F164 C263F1          JNZ    $-1
F167 00              NOP
F168 00              NOP
F169 3E45            MVI    A,45H   ;SCOT AL
F16B D3D8            OUT    0D8H    ;
               ;INT. SCRIERE PRIMA ZONA HEADER
F16D 1E43            MVI    E,43H
F16F 1D              DCR    E
F170 C26FF1          JNZ    $-1
F173 00              NOP
F174 00              NOP
F175 3E01            MVI    A,1     ;INR.SECT.
F177 32F0F2          STA    ADINS   ;
F17A 3EC7            MVI    A,0C7H  ;MARK HEADER
F17C 32DFE7          STA    AMKCK   ;
F17F 3EFE      SECT: MVI    A,0FEH  ;
F181 32CFE7          STA    AMRKF   ;
F184 2AF6F2          LHLD   LT7B    ;NR. SECT.
F187 23              INX    H       ;
F188 22F6F2          SHLD   LT7B    ;
F18B 7E              MOV    A,M     ;
F18C 3202E7          STA    ANRSF   ;
               ;PREGATIRE DMA ZONA DMA HEADER
F18F 11D9E7          LXI    D,FPREK
F192 011180          LXI    B,8011H
F195 CDD4EE          CALL   DMA0
F198 11C9E7          LXI    D,FRMDT+3
F19B 010C80          LXI    B,800CH
F19E CDE2EE          CALL   DMA2
F1A1 11C8E7          LXI    D,FRMDT+2
F1A4 010580          LXI    B,8005H
F1A7 CDF0EE          CALL   DMA3
               ;VALIDARE CRC
                     PUN    PCSFL,NRZDTA,1
F1AA+DBDE            IN     PCSFL
F1AC+F601            ORI    NRZDTA
F1AE+D3DE            OUT    PCSFL
               ;VALIDARE TRECERE MARK BI
                     PUN    PCSFL,NDRQ,1
F1B0+DBDE            IN     PCSFL
F1B2+F608            ORI    NDRQ
F1B4+D3DE            OUT    PCSFL
                     PUN    PCSFL,NDRQ,0
F1B6+DBDE            IN     PCSFL
F1B8+E6F7            ANI    0-1-   NDRQ
F1BA+D3DE            OUT    PCSFL
F1BC 3EC5            MVI    A,0C5H  ;LANS.DMA
F1BE D3D8            OUT    0D8H    ;
               ;INT. DEPASIRE TC CAN2
F1C0 1E34            MVI    E,34H
F1C2 1D              DCR    E
F1C3 C2C2F           JNZ    $-
F1C6 3E45            MVI    A,45H   ;SCOT AL
F1C8 D3D8            OUT    0D8H
               ;INVALIDARE CRC
```



```
                     PUN    PCSFL,NRZDTA,0
F1CA+DBDE            IN     PCSFL
F1CC+E6FE            ANI    0-1-   NRZDTA
F1CE+D3DE            OUT    PCSFL
F1D0 3EFB            MVI    A,0FBH  ;MARK BLOC
F1D2 32CFE7          STA    AMRKF   ;
               ; DMA ZONA MARK BLOC,INCEPUT BLOC
F1D5 11F3F2          LXI    D,ABLOC
F1D8 010380          LXI    B,8003H
F1DB CDF0EE          CALL   DMA3
F1DE 11D9E7          LXI    D,FPREK
F1E1 010A80          LXI    B,800AH
F1E4 CDD4EE          CALL   DMA0
F1E7 11C9E7          LXI    D,FRMDT+3
F1EA 010780          LXI    B,8007H
F1ED CDE2EE          CALL   DMA2
               ;INT.HEADER BLOC
F1F0 1E08      CONT:   MVI E,8H
F1F2 1D              DCR    E
F1F3 C2F2F1          JNZ    $-1
F1F6 3E00            MVI    A,0
F1F8 00              NOP
F1F9 7F              MOV    A,A
               ;VALIDARE TRECERE MARK BLOC
                     PUN    PCSFL,NDRQ,1
F1FA+DBDE            IN     PCSFL
F1FC+F608            ORI    NDRQ
F1FE+D3DE            OUT    PCSFL
                     PUN    PCSFL,NDRQ,0
F200+DBDE            IN     PCSFL
F202+E6F7            ANI    0-1-   NDRQ
F204+D3DE            OUT    PCSFL
F206 3EC5            MVI    A,0C5H  ;LANS.DMA
F208 D3D8            OUT    0D8H    ;
               ;INT DEPASIRE TC CAN2
F20A 1E1C            MVI    E,1CH
F20C 1D              DCR    E
F20D C20CF2          JNZ    $-1
F210 00              NOP
F211 00              NOP
F212 3E45            MVI    A,45H   ;SCOT AL
F214 D3D8            OUT    0D8H    ;
               ;INT.LUNGIME BLOC
F216 1EF3            MVI    E,0F3H
F218 1D              DCR    E
F219 C218F2          JNZ    $-1
F21C 1EFF            MVI    E,0FFH
F21E 1D              DCR    E
F21F C21EF2          JNZ    $-1
F222 1E06            MVI    E,06H
F224 1D              DCR    E
F225 C224F2          JNZ    $-1
F228 00              NOP
F229 00              NOP
               ;VALIDARE CRC
                     PUN    PCSFL,NRZDTA,1
F22A+DBDE            IN     PCSFL
F22C+F601            ORI    NRZDTA
F22E+D3DE            OUT    PCSFL
               ;PREGATIRE DMA FINAL BLOC
F230 11B7E7          LXI    D,FRMCK+1
F233 010880          LXI    B,8008H
F236 CDD4EE          CALL   DMA0
F239 11F3F2          LXI    D,ABLOC
F23C 010380          LXI    B,8003H
F23F CDE2EE          CALL   DMA2
F242 11C8E7          LXI    D,FRMDT+2
F245 010580          LXI    B,8005H
```

```
 F248 CDF0EE              CALL    DMA3
 F24B 3EC5                MVI     A,0C5H  ;LANS DMA
 F24D D3D8                OUT     0D8H    ;
                 ;INT.DEPASIRE TC CAN2
 F24F 1E11                MVI     E,11H
 F251 1D                  DCR     E
 F252 C251F2              JNZ     $-1
 F255 00                  NOP
 F256 00                  NOP
 F257 3E45                MVI     A,45H   ;SCOT AL
 F259 D3D8                OUT     0D8H
                 ;INVALIDARE CRC
                          PUN     PCSFL,NRZDTA,0
 F25B+DBDE                IN      PCSFL
 F25D+E6FE                ANI     0-1-    NRZDTA
 F25F+D3DE                OUT     PCSFL
 F261 3AF0F2              LDA     ADINS   ;INR.NR.SECT
 F264 3C                  INR     A       ;
 F265 32F0F2              STA     ADINS   ;
 F268 FE1B                CPI     1BH     ;
 F26A CA7EF2              JZ      ENDY    ;
                 ;PREG DATE BLOC URMATOR
 F26B CD9CF2              CALL    CD
                 ;INT.BLOC-HEADER
 F270 1E38                MVI     E,38H
 F272 1D                  DCR     E
 F273 C272F2              JNZ     $-1
 F276 C379F2              JMP     $+3
 F279 3E00                MVI     A,0
 F27B C37FF1              JMP     SECT
 F27E CDCCF2     ENDY:    CALL    TFCIX
 F281 1E01                MVI     E,1
 F283 1D                  DCR     E
 F284 C283F2              JNZ     $-1
                          PUN     PCSFL,NRZDTA,0
 F287+DBDE                IN      PCSFL
 F289+E6FE                ANI     0-1-    NRZDTA
 F28B+D3DE                OUT     PCSFL
                          PUN     PCSFL,NDRQ,1
 F28D+DBDE                IN      PCSFL
 F28F+F608                ORI     NDRQ
 F291+D3DE                OUT     PCSFL
                          PUN     PCFL,NWRIT,1
 F293+DBDC                IN      PCFL
 F295+F610                ORI     NWRIT
 F297+D3DC                OUT     PCFL
 F299 C3DBE9              JMP     FINAL
                 ;STABILIRE CIMP DATE
 F29C 3A10E8     CD:      LDA ADRCC
 F29F E640                ANI     40H
 F2A1 C2B6F2              JNZ     CDFA
 F2A4 21EFF2     CDFN:    LXI     H,ADTSCT
 F2A7 11F3F2              LXI     D,ABLOC
 F2AA CDC5F2              CALL    MOV3
 F2AD 00                  NOP
 F2AE 00                  NOP
 F2AF 00                  NOP
 F2B0 00                  NOP
 F2B1 00                  NOP
 F2B2 00                  NOP
 F2B3 00                  NOP
 F2B4 00                  NOP
 F2B5 C9                  RET
 F2B6 2AF1F2     CDFA:    LHLD    AADFA
 F2B9 11F3F2              LXI     D,ABLOC
 F2BC CDC5F2              CALL    MOV3
 F2BF 23                  INX     H
 F2C0 23                  INX     H
```



```
 F2C1 22F1F2              SHLD    AADFA
 F2C4 C9                  RET
 F2C5 7E         MOV3:    MOV     A,M
 F2C6 12                  STAX    D
 F2C7 13                  INX     D
 F2C8 12                  STAX    D
 F2C9 13                  INX     D
 F2CA 12                  STAX    D
 F2CB C9                  RET
 F2CC DBDD       TFCIX:   IN      PSFL
 F2CE E608                ANI     INDX
 F2D0 C2CCF2              JNZ     TFCIX
 F2D3 DBDD       ASTEP:   IN      PSFL
 F2D5 E608                ANI     INDX
 F2D7 CAD3F2              JZ      ASTEP
 F2DA C9                  RET
 F2DB CDE2F2     TSTWPF:  CALL    INWPR
 F2DE C2E7F2              JNZ     ERWPF
 F2E1 C9                  RET
 F2E2 DBDD       INWPR:   IN      PSFL
 F2E4 E640                ANI     WPR
 F2E6 C9                  RET
 F2E7 3E20       ERWPF:   MVI     A,20H
 F2E9 322CEF              STA     ERRYTE
 F2EC C3C1EE              JMP     FIXPAR
 F2EF E5         ADTSCT:  DB      0E5H
 F2F0            ADINS:   DS      1
 F2F1            AADFA:   DS      2
 F2F3            ABLOC:   DS      3
 F2F6            LT7B:    DS      2
                 ;
                 ;
                 ;
                 ;
                 ;FORMATARE IN DUBLADENSITATE
                 ;
                 ;
                 ;
                 ;
                 ;FORMATARE SD NORMALA SAU ALEATOARE
 F2F8 3A13E8     FORTM:   LDA ADNRP
 F2FB FE4D                CPI     77D
 F2FD D2ABE9              JNC     ERPRG
 F300 CDDBF2              CALL    TSTWPF
 F303 CD3FE7              CALL    INICO
 F306 3A10E8              LDA     ADRCC
 F309 E640                ANI     40H
 F30B C21EF3              JNZ     FALEADD
                 ;COMPLETARE TABEL FORMATARE NORMALA
 F30E 213EEF     FNORMDD:         LXI H,T7B+1
 F311 3E01                MVI     A,1
 F313 77         CT7BDD:  MOV     M,A
 F314 23                  INX     H
 F315 3C                  INR     A
 F316 FE1B                CPI     1BH
 F318 C213F3              JNZ     CT7BDD
 F31B C330F3              JMP     FORMDD
                 ; TABEL FORMATARE ALEATOARE LA T7B
 F31E 2A15E8     FALEADD:         LHLD    ABLAD
 F321 23                  INX     H
 F322 2265F5              SHLD    AADFADD
 F325 2B                  DCX     H
 F326 EB                  XCHG
 F327 013EEF              LXI     B,T7B+1
 F32A CD30F3              CALL    MOV2DD
 F32D C330F3              JMP     FORMDD
 F330 2E1A       MOV2DD:  MVI     L,1AH
 F332 1A         RETADD:  LDAX    D
```

```
F333 02             STAX    B
F334 13             INX     D
F335 13             INX     D
F336 03             INX     B
F337 2D             DCR     L
F338 C232F3         JNZ     REIADD
F33B C9             RET
                ;FORMATARE PISTA
F33C 3A13E8     FORMDD: LDA ADNRP
F33F FE00           CPI     0
F341 CA55F3         JZ      ALPHADD
F344 3A13E8         LDA     ADNRP
F347 FE2B           CPI     43D
F349 D433EB         CNC     LCRNT
F34C CDB3EA         CALL    HDSTP
F34F CD55EA     BETADD: CALL    HLOD
F352 C364F3         JMP     FORDD
F355 DBDD       ALPHADD:        IN      PSFL
F357 E604           ANI     TRK0
F359 C24FF3         JNZ     BETADD
F35C 0E01           MVI     C,1
F35E CDC3EA         CALL    OUTST
F361 C355F3         JMP     ALPHADD
F364 3A13E8     FORDD:  LDA     ADNRP
F367 3285F5         STA     ANRPFD
F36A 210DEF         LXI     H,T7B
F36D 226BF5         SHLD    LT7BDD
F370 3E14           MVI     A,14H   ;MARK INDEX
F372 3295F5         STA     ADCLKDD+0CH     ;
F375 3296F5         STA     ADCLKDD+0DH     ;
F378 3297F5         STA     ADCLKDD+0EH     ;
F37B 3EFC           MVI     A,0FCH  ;
F37D 3284F5         STA     ANRPFD-1        ;
F380 3EC2           MVI     A,0C2H  ;
F382 3281F5         STA     ANRPFD-4        ;
F385 3282F5         STA     ANRPFD-3        ;
F388 3283F5         STA     ANRPFD-2        ;
F38B 3E0C           MVI     A,ZCSFLM
F38D D3DE           OUT     PCSFL
F38F CD81F5         CALL    CBDD    ;PREG DATE PRIM BLOC
                ;PREGATIRE DMA INCEPUT PISTA
F392 1189F5         LXI     D,ADCLKDD
F395 010480         LXI     B,8004H
F398 CDD4EE         CALL    DMA0
F39B 116DF5         LXI     D,ADDTADD
F39E 010480         LXI     B,8004H
F3A1 CDE2EE         CALL    DMA2
F3A4 3E45           MVI     A,45H
F3A6 D3D8           OUT     0D8H
F3A8 CDC0F2         CALL    TFCIX
                    PUN     PCSFL,NDRO,0
F3AB+DBDE           IN      PCSFL
F3AD+E6F7           ANI     0-1-    NDRO
F3AF+D3DE           OUT     PCSFL
                    PUN     PCFL,NWRIT,0
F3B1+DBDC           IN      PCFL
F3B3+E6EF           ANI     0-1-    NWRIT
F3B5+D3DC           OUT     PCFL
                ;INTIRZIERE SCR. GAP MARK INDEX
F3B7 1E89           MVI     E,89H
F3B9 1D             DCR     E
F3BA C2B9F3         JNZ     $-1
F3BD 7F             MOV     A,A
                ;PREGATIRE DMA ZONA MARK INDEX
F3BE 1189F5         LXI     D,ADCLKDD
F3C1 011480         LXI     B,8014H
F3C4 CDD4EE         CALL    DMA0
F3C7 1175F5         LXI     D,ADDTADD+8
```



```
F3CA 011080         LXI     B,8010H
F3CD CDE2EE         CALL    DMA2
F3D0 116DF5         LXI     D,ADDTADD
F3D3 010480         LXI     B,8004H
F3D6 CDF0EE         CALL    DMA3
F3D9 3EC5           MVI     A,0C5H  ;LANS DMA
F3DB D3D8           OUT     0D8H
                ;INT. DEPASIRE TC CAN2
F3DD 1E19           MVI     E,19H
F3DF 1D             DCR     E
F3E0 C2DFF3         JNZ     $-1
F3E3 3E00           MVI     A,0
F3E5 3E00           MVI     A,0
F3E7 3E45           MVI     A,45H   ;SCRT AL
F3E9 D3D8           OUT     0D8H    ;
                ;INT. SCRIERE PRIMA ZONA HEADER
F3EB 1E3B           MVI     E,3BH
F3ED 1D             DCR     E
F3EE C2EDF3         JNZ     $-1
F3F1 00             NOP
F3F2 00             NOP
F3F3 00             NOP
F3F4 00             NOP
                ;STAB.LOC.INR.SECTDD.
F3F5 3E01           MVI     A,1
F3F7 3264F5         STA     ADINSDD ;
F3FA 3E0A           MVI     A,0AH   ;MARK HEADER
F3FC 3295F5         STA     ADCLKDD+0CH     ;
F3FF 3296F5         STA     ADCLKDD+0DH     ;
F402 3297F5         STA     ADCLKDD+0EH
F405 3EFE       SECTDD: MVI     A,0FEH  ;
F407 3284F5         STA     ANRPFD-1
F40A 3EA1           MVI     A,0A1H  ;
F40C 3281F5         STA     ANRPFD-4
F40F 3282F5         STA     ANRPFD-3
F412 3283F5         STA     ANRPFD-2
                ;STAB. NR. SECTDD.
F415 2A6BF5         LHLD    LT7BDD
F418 23             INX     H       ;
F419 226BF5         SHLD    LT7BDD  ;
F41C 7E             MOV     A,M     ;
F41D 3287F5         STA     ANRSFD  ;
                ;PREGATIRE DMA ZONA DMA HEADER
F420 1189F5         LXI     D,ADCLKDD
F423 011D80         LXI     B,801DH
F426 CDD4EE         CALL    DMA0
F429 1175F5         LXI     D,ADDTADD+8
F42C 011580         LXI     B,8015H
F42F CDE2EE         CALL    DMA2
F432 116DF5         LXI     D,ADDTADD
F435 010080         LXI     B,8008H
F438 CDF0EE         CALL    DMA3
                ;VALIDARE CRC
                    PUN     PCSFL,NRZDTA,1
F43B+DBDE           IN      PCSFL
F43D+F601           ORI     NRZDTA
F43F+D3DE           OUT     PCSFL
                ;VALIDARE TRECERE MARK BI
                    PUN     PCSFL,NDRO,1
F441+DBDE           IN      PCSFL
F443+F608           ORI     NDRO
F445+D3DE           OUT     PCSFL
                    PUN     PCSFL,NDRO,0
F447+DBDE           IN      PCSFL
F449+E6F7           ANI     0-1-    NDRO
F44B+D3DE           OUT     PCSFL
F44D 3EC5           MVI     A,0C5H  ;LANS.DMA
F44F D3D8           OUT     0D8H
```

```
                ;INT. DEPASIRE TC CAN2
F451 1E23               MVI     E,23H
F453 1D                 DCR     E
F454 C253F4             JNZ     $-1
F457 3E45               MVI     A,45H   ;SCOT AL
F459 D3D8               OUT     0D8H    ;
                ;INVALIDARE CRC
                        PUN     PCSFL,NRZDTA,0
F45B+DBDE               IN      PCSFL
F45D+E6FE               ANI     0-1-    NRZDTA
F45F+D3DE               OUT     PCSFL
                ;MARK BLOC
F461 3EFB               MVI     A,0FBH
F463 3284F5             STA     ANRPFD-1
                ; DMA ZONA MARK BLOC,INCEPUT BLOC
F466 1167F5             LXI     D,ABLOCDD
F469 010480             LXI     B,8004H
F46C CDF0EE             CALL    DMA3
F46F 1189F5             LXI     D,ADCLKDD
F472 011480             LXI     B,8014H
F475 CDD4EE             CALL    DMA0
F478 1175F5             LXI     D,ADDTADD+8
F47B 011080             LXI     B,8010H
F47E CDE2EE             CALL    DMA2
                ;INT.HEADER-BLOC
F481 1E09       CONTDD:  MVI E,9H
F483 1D                 DCR     E
F484 C283F4             JNZ     $-1
F487 3E00               MVI     A,0
F489 00                 NOP
F48A 00                 NOP
F48B 7F                 MOV     A,A
                ;VALIDARE TRECERE MARK BLOC
                        PUN     PCSFL,NDR0,1
F48C+DBDE               IN      PCSFL
F48E+F608               ORI     NDR0
F490+D3DE               OUT     PCSFL
                        PUN     PCSFL,NDR0,0
F492+DBDE               IN      PCSFL
F494+E6F7               ANI     0-1-    NDR0
F496+D3DE               OUT     PCSFL
F498 3EC5               MVI     A,0C5H  ;LANS.DMA
F49A D3D8               OUT     0D8H    ;
                ;INT DEPASIRE TC CAN2
F49C 1E19               MVI     E,19H
F49E 1D                 DCR     E
F49F C29EF4             JNZ     $-1
F4A2 00                 NOP
F4A3 00                 NOP
F4A4 3E45               MVI     A,45H   ;SCOT AL
F4A6 D3D8               OUT     0D8H    ;
                ;INT.LUNGIME BLOC
F4A8 1EF3               MVI     E,0F3H
F4AA 1D                 DCR     E
F4AB C2AAF4             JNZ     $-1
F4AE 1EFF               MVI     E,0FFH
F4B0 1D                 DCR     E
F4B1 C2B0F4             JNZ     $-1
F4B4 1E0A               MVI     E,0AH
F4B6 1D                 DCR     E
F4B7 C2B6F4             JNZ     $-1
F4BA 7F                 MOV     A,A
F4BB 7F                 MOV     A,A
F4BC 00                 NOP
                ;VALIDARE CRC
                        PUN     PCSFL,NRZDTA,1
F4BD+DBDE               IN      PCSFL
F4BF+F601               ORI     NRZDTA
```

```
F4C1+D3DE               OUT     PCSFL
                ;PREGATIRE DMA FINAL BLOC
F4C3 1189F5             LXI     D,ADCLKDD
F4C6 010C80             LXI     B,800CH
F4C9 CDD4EE             CALL    DMA0
F4CC 1167F5             LXI     D,ABLOCDD
F4CF 010480             LXI     B,8004H
F4D2 CDE2EE             CALL    DMA2
F4D5 116DF5             LXI     D,ADDTADD
F4D8 010880             LXI     B,8008H
F4DB CDF0EE             CALL    DMA3
F4DE 3EC5               MVI     A,0C5H  ;LANS DMA
F4E0 D3D8               OUT     0D8H    ;
                ;INT.DEPASIRE TC CAN2
F4E2 1E08               MVI     E,08H
F4E4 1D                 DCR     E
F4E5 C2E4F4             JNZ     $-1
F4E8 3E00               MVI     A,00
F4EA 3E45               MVI     A,45H   ;SCOT AL
F4EC D3D8               OUT     0D8H
                ;INVALIDARE CRC
                        PUN     PCSFL,NRZDTA,0
F4EE+DBDE               IN      PCSFL
F4F0+E6FE               ANI     0-1-    NRZDTA
F4F2+D3DE               OUT     PCSFL
                ;INCREMENTARE NUMAR SECTOR
F4F4 3A64F5             LDA     ADINSDD
F4F7 3C                 INR     A       ;
F4F8 3264F5             STA     ADINSDD ;
F4FB FE1B               CPI     1BH     ;
F4FD CA11F5             JZ      ENDVDD  ;
                ;PREGATIRE DATE BLOC URMATOR
F500 CD31F5             CALL    CDDD
                ;INT.BLOC-HEADER
F503 1E03               MVI     E,3H
F505 1D                 DCR     E
F506 C205F5             JNZ     $-1
F509 7F                 MOV     A,A
F50A 7F                 MOV     A,A
F50B 00                 NOP
F50C 00                 NOP
F50D 00                 NOP
F50E C305F4             JMP     SECTDD
F511 CDCCF2     ENDVDD: CALL    TPCIX
F514 1E02               MVI     E,2
F516 1D                 DCR     E
F517 C216F5             JNZ     $-1
F51A E601               ANI     01
                        PUN     PCSFL,NRZDTA,0
F51C+DBDE               IN      PCSFL
F51E+E6FE               ANI     0-1-    NRZDTA
F520+D3DE               OUT     PCSFL
                        PUN     PCSFL,NDR0,1
F522+DBDE               IN      PCSFL
F524+F608               ORI     NDR0
F526+D3DE               OUT     PCSFL
                        PUN     PCFL,NURIT,1
F528+DBDC               IN      PCFL
F52A+F610               ORI     NURIT
F52C+D3DC               OUT     PCFL
F52E C3BBE9             JMP     FINAL
                ;STABILIRE CIMP DATE
F531 3A10E8     CDDD:    LDA ADRCC
F534 E640               ANI     40H
F536 C24BF5             JNZ     CDFADD
F539 2163F5     CDFNDD: LXI     H,ADTSCTDD
F53C 1167F5             LXI     D,ABLOCDD
F53F CD5AF5             CALL    MOV3DD
```

```
F542 00                 NOP
F543 00                 NOP
F544 00                 NOP
F545 00                 NOP
F546 00                 NOP
F547 00                 NOP
F548 00                 NOP
F549 00                 NOP
F54A C9                 RET
F54B 2A65F5    COFADD:  LHLD    AADFADD
F54E 1167F5             LXI     D,ABLOCDD
F551 CD5AF5             CALL    MOV3DD
F554 23                 INX     H
F555 23                 INX     H
F556 2265F5             SHLD    AADFADD
F559 C9                 RET
F55A 7E        MOV3DD:  MOV     A,M
F55B 12                 STAX    D
F55C 13                 INX     D
F55D 12                 STAX    D
F55E 13                 INX     D
F55F 12                 STAX    D
F560 13                 INX     D
F561 12                 STAX    D
F562 C9                 RET
F563 E5        ADTSCTDD:        DB      0E5H
F564           ADINSDD:         DS      1
F565           AADFADD:         DS      2
F567           ABLOCDD:         DS      4
F56B           LT7BDD: DS       2
               ADDTADD:         REPT    8D
                        DB      04EH
                        ENDM
F56D+4E                 DB      04EH
F56E+4E                 DB      04EH
F56F+4E                 DB      04EH
F570+4E                 DB      04EH
F571+4E                 DB      04EH
F572+4E                 DB      04EH
F573+4E                 DB      04EH
F574+4E                 DB      04EH
                        REPT    12D
                        DB      00
                        ENDM
F575+00                 DB      00
F576+00                 DB      00
F577+00                 DB      00
F578+00                 DB      00
F579+00                 DB      00
F57A+00                 DB      00
F57B+00                 DB      00
F57C+00                 DB      00
F57D+00                 DB      00
F57E+00                 DB      00
F57F+00                 DB      00
F580+00                 DB      00
                        REPT    3
                        DB      0A1H
                        ENDM
F581+A1                 DB      0A1H
F582+A1                 DB      0A1H
F583+A1                 DB      0A1H
F584 FE                 DB      0FEH
F585 00                 DB      00
F586 00                 DB      00
F587 01                 DB      01
F588 01                 DB      01
               ABDCLKDD:        REPT    12D
```



```
                        DB      0FFH
                        ENDM
F589+FF                 DB      0FFH
F58A+FF                 DB      0FFH
F58B+FF                 DB      0FFH
F58C+FF                 DB      0FFH
F58D+FF                 DB      0FFH
F58E+FF                 DB      0FFH
F58F+FF                 DB      0FFH
F590+FF                 DB      0FFH
F591+FF                 DB      0FFH
F592+FF                 DB      0FFH
F593+FF                 DB      0FFH
F594+FF                 DB      0FFH
                        REPT    3
                        DB      0AH
                        ENDM
F595+0A                 DB      0AH
F596+0A                 DB      0AH
F597+0A                 DB      0AH
                        REPT    14D
                        DB      0FFH
                        ENDM
F598+FF                 DB      0FFH
F599+FF                 DB      0FFH
F59A+FF                 DB      0FFH
F59B+FF                 DB      0FFH
F59C+FF                 DB      0FFH
F59D+FF                 DB      0FFH
F59E+FF                 DB      0FFH
F59F+FF                 DB      0FFH
F5A0+FF                 DB      0FFH
F5A1+FF                 DB      0FFH
F5A2+FF                 DB      0FFH
F5A3+FF                 DB      0FFH
F5A4+FF                 DB      0FFH
F5A5+FF                 DB      0FFH
F585 =                  ANRPFD  EQU     ADDTADD+18H
F587 =                  ANRSFD  EQU     ADDTADD+1AH
F5A6                    DS      100H
F6A6 =         STACK    EQU     $
F6A6                    DS      100H
F7A6 =         MEMORY   EQU     $
F7A6                    END
```

PROGRAM DE ÎNCĂRCARE LA PUNEREA SUB TENSIUNE

```
INCARCAREA SISTEMULUI' MACRO-80 3.36   17-Mar-80        PAGE    I-1


                                   ;DATA: 05.08.86
0000                                      SBASE   EQU     00000H
0078                                      BASE    EQU     78H

                                          ORG     SBASE

0000'     C3 014F'                        JMP     SHO

                                   .Z80
                                          ;INITIALIZARE CTC/UC
0003'     3E 05                    INIPER: LD     A,05H   ;COMANDA CANAL 0
0005'     D3 F4                           OUT     (0F4H),A
0007'     3E 08                           LD      A,08H   ;CONSTANTA CANAL 0
0009'     D3 F4                           OUT     (0F4H),A        ;1200 BAUD
                                          ;INITIALIZARE SIO/UC
000B'     3E 18                           LD      A,18H   ;INITIALIZARE CAN. A
000D'     D3 FA                           OUT     (0FAH),A
000F'     D3 FA                           OUT     (0FAH),A
0011'     3E 14                           LD      A,14H   ;ADRESARE WR4
0013'     D3 FA                           OUT     (0FAH),A        ;INITIALIZARE INT.
0015'     3E 4C                           LD      A,4CH   ;RATA DATE X 16=CEAS
0017'     D3 FA                           OUT     (0FAH),A        ;2 BITI STOP
0019'     3E 35                           LD      A,35H   ;ADRESARE WR5
001B'     D3 FA                           OUT     (0FAH),A        ;INITIALIZARE ERORI
001D'     3E EA                           LD      A,0EAH  ;8BITI/CAR
001F'     D3 FA                           OUT     (0FAH),A        ;VALID. TRANSM.
0021'     3E 13                           LD      A,13H   ;ADRESARE WR3
0023'     D3 FA                           OUT     (0FAH),A
0025'     3E C1                           LD      A,0C1H  ;8BITI/CAR
0027'     D3 FA                           OUT     (0FAH),A        ;VALID REC.
                                          ;INITIALIZARE CTC/CDF
0029'     ED 5E                           IM      2       ;Z80 IN  MODUL 2
002B'     21 F000'                        LD      HL,INI  ;HL=ADR. TAB. ADRESE
002E'     7C                              LD      A,H
002F'     ED 47                           LD      I,A
0031'     DD 21 F008'                     LD      IX,IT7  ;INCARCARE TAB. ADR.
0035'     DD 22 F000'                     LD      (INI),IX
0039'     DD 21 F00D'                     LD      IX,IT6
003D'     DD 22 F005'                     LD      (INI+5),IX
0041'     DD 21 F012'                     LD      IX,IT5
0045'     DD 22 F00A'                     LD      (INI+10),IX
0049'     DD 21 F017'                     LD      IX,IT4
004D'     DD 22 F01F                      LD      (0F010H+15),IX
0051'     3E 10                           LD      A,10H   ;VECTOR INTR.
0053'     D3 E4                           OUT     (0E4H),A
0055'     3E D7                           LD      A,0D7H  ;COMD. CAN. 0
0057'     D3 E4                           OUT     (0E4H),A
0059'     3E 01                           LD      A,1     ;CONST. CAN 0
005B'     D3 E4                           OUT     (0E4H),A
005D'     3A 0007                         LD      A,(0D7H)        ;COMD. CAN. 1
0060'     D3 E5                           OUT     (0E5H),A
0062'     3E 01                           LD      A,1     ;CONST. CAN. 1
0064'     D3 E5                           OUT     (0E5H),A
0066'     3E D7                           LD      A,0D7H  ;COMD. CAN. 2
0068'     D3 E6                           OUT     (0E6H),A
006A'     3E 01                           LD      A,1     ;CONST. CAN. 2
006C'     D3 E6                           OUT     (0E6H),A
006E'     3E D7                           LD      A,0D7H  ;COMD.CAN. 3
0070'     D3 E7                           OUT     (0E7H),A
0072'     3E 01                           LD      A,1     ;CONST.CAN. 3
0074'     D3 E7                           OUT     (0E7H),A
0076'     C9                              RET
```

```
'INCARCAREA SISTEMULUI' MACRO-80 3.36   17-Mar-80   PAGE 1-2

                                ;IESIRE CARACTER
0077'   DB FA           CO:     IN      A,(0FAH)
0079'   CB 57                   BIT     2,A
007B'   28 FA                   JR      Z,CO
007D'   79                      LD      A,C
007E'   D3 F8                   OUT     (0F8H),A
0080'   C9                      RET
                                ;INTRARE CARACTER
0081'   DB FA           CI:     IN      A,(0FAH)
0083'   CB 47                   BIT     0,A
0085'   28 FA                   JR      Z,CI
0087'   DB F8                   IN      A,(0F8H)
0089'   C9                      RET
                                ;TEST CONSOLA
008A'   DB FA           CSTS:   IN      A,(0FAH)
008C'   CB 47                   BIT     0,A
008E'   28 03                   JR      Z,NOCH
0090'   3E FF                   LD      A,0FFH
0092'   C9                      RET
0093'   AF              NOCH:   XOR     A
0094'   C9                      RET

                        .8080

                                ORG     SBASE+008H

00C8'                   TOS:
00C0'                   USER    EQU     TOS-8
00C8'   EE              ELOC:   DB      0EEH
00C9'   DD              DLOC:   DB      0DDH
00CA'   CC              CLOC:   DB      0CCH
00CB'   BB              BLOC:   DB      0BBH
00CC'   00                      DB      0
00CD'   FE              ILOC:   DB      0FEH
00CE'   FF              FLOC:   DB      0FFH
00CF'   AA              ALOC:   DB      0AAH
00D0'   C0                      DB      LOW USER
00D1'   00              SLOC:   DB      0
00D2'                   EXIT:
00D2'   F3                      DI
00D3'   D1                      POP     D
00D4'   C1                      POP     B
00D5'   F1                      POP     PSW
00D6'   D3 FC                   OUT     0FCH
00D8'   F1                      POP     PSW
00D9'   E1                      POP     H
00DA'   F9                      SPHL
00DB'   21 1234                 LXI     H,1234H
00DC'                   LLOC    EQU     $-2
00DD'                   HLOC    EQU     $-1
00DE'   FB                      EI
00DF'   C3 6789                 JMP     6789H
00E1'                   PLOC    EQU     $-1
00E2'   0000            TLOC:   DW      0
00E4'   00                      DB      0
00E5'   0000                    DW      0
00E7'   00                      DB      0
                        ;EXTENSIBLE I/O ENTRY POINTS
00E8'                   XTBL:
00E8'   C3 0000         CILOC:  JMP     0
00EB'   C3 0000         COLOC:  JMP     0
00EE'   C3 0000         R1LOC:  JMP     0
00F1'   C3 0000         R2LOC:  JMP     0
00F4'   C3 0000         P1LOC:  JMP     0
```

```
'INCARCAREA SISTEMULUI' MACRO-80 3.36   PAGE 1-3

00F7'   C3 0000         P2LOC:  JMP     0
00FA'   C3 0000         L1LOC:  JMP     0
00FD'   C3 0000         CSLOC:  JMP     0

                        ;INCARCAREA SISTEMULUI DE OPERARE
0100'                   ISISII:
0100'   FB                      EI
0101'   11 0117'                LXI     D,ADIOP
0104'   7B                      MOV     A,E
0105'   D3 79                   OUT     BASE+1
0107'   7A                      MOV     A,D
0108'   D3 7A                   OUT     BASE+2
010A'   DB 78           WAITIT: IN      BASE
010C'   E6 04                   ANI     4
010E'   CA 010A'                JZ      WAITIT
0111'   DB 78                   IN      BASE
0113'   0F                      RRC
0114'   C3 3000                 JMP     3000H
0117'   80              ADIOP:  DB      80H
0118'   04              ORDIN:  DB      04H
0119'   1A              NRSCT:  DB      1AH
011A'   00              NRPST:  DB      00H
011B'   01              PRMSCT: DB      01H
011C'   3000            ADTR:   DW      3000H
                        ;PROGRAMARE PERIFERIE
011E'                   IODEV:
011E'   CD 0003'                CALL    INIPER
                        ;STABILESTE OCTET I/E
0121'   21 0003                 LXI     H,IOBYT
0124'   16 41                   MVI     D,41H
0126'   72                      MOV     M,D
                        ;CONFIGURATIA INITIALA
0127'   2E 06                   MVI     L,06
0129'   72                      MOV     M,D
                        ;TYPE SIGN-ON FOR RAM MONITOR
012A'   21 0139'                LXI     H,VERS
012D'   06 11                   MVI     B,LVER
012F'   4E              LP020:  MOV     C,M
0130'   CD 0077'                CALL    CO
0133'   23                      INX     H
0134'   05                      DCR     B
0135'   C2 012F'                JNZ     LP020
0138'   C9                      RET
                        ;MDS MONITOR SIGN-ON MESSAGE
0139'   0D 0A           VERS:   DB      0DH,0AH
013B'   4D 4F 44 55     CARA:   DB      'MODULAR 1986'
013F'   4C 41 52 20
0143'   20 31 39 38
0147'   36
0148'   0D 0A           CRLF:   DB      0DH,0AH
0011                    LVER    EQU     $-VERS
0886                    DATE    EQU     0886H
0004                    MEMTP   EQU     0004
0000                    RESET   EQU     0
0003                    IOBYT   EQU     03
014A'   00              INIT:   DB      0
014B'   0886                    DW      DATE
014D'   F3              SH0:    DI
                        ;TEST DIMENSIUNE MEMORIE
014E'   21 1800                 LXI     H,SBASE+1800H
0151'   CD 01A6'                CALL    LOOP
0154'   2B                      DCX     H
0155'   7C                      MOV     A,H
```

```
0156'   D6 10                   SUI     10H
0158'   67                      MOV     H,A
0159'   22 0004                 SHLD    MEMTP
015C'   22 0014                 SHLD    MEMTP+10H
                        ;COPIAZA IN LOCATII DEDICATE
015F'   01 00C8'                LXI     B,TOS
0162'   69                      MOV     L,C
0163'   F9                      SPHL
0164'   0A              SH1:    LDAX B
0165'   77                      MOV     M,A
0166'   0C                      INR     C
0167'   2C                      INR     L
0168'   C2 0164'                JNZ     SH1
016B'   2E 01                   MVI     L,LOW SLOC
016D'   74                      MOV     M,H
016E'   35                      DCR     M
                        ;INITIALIZARI PAGINA 0
0066                    ANMI    EQU     66H
016F'   3E C3                   MVI     A,0C3H
0171'   32 0000                 STA     RESET
0174'   32 0066                 STA     ANMI
0177'   21 FF0F                 LXI     H,0FF0FH
017A'   22 0001                 SHLD    RESET+1
017D'   22 0067                 SHLD    ANMI+1
                        ;INITIALIZARI PERIFERICE
0180'   CD 011E'                CALL    IODEV
                        ;COPIAZA PROM-URILE
0183'   2A 0004                 LHLD    MEMTP
0186'   4D                      MOV     C,L
0187'   44                      MOV     B,H
0188'   03                      INX     B
0189'   11 01C1'                LXI     D,TOT
018C'   21 1800                 LXI     H,1800H
018F'   CD 01B2'                CALL    COPY
                        ;TEST INTRARE SO/MON
0192'                   INTRARE:
0192'   CD F01F'                CALL    INICDF
0195'   DB 78                   IN      BASE
0197'   E6 08                   ANI     08H
0199'   CA F01C'                JZ      MONIT
019C'   DB 78                   IN      BASE
019E'   E6 01                   ANI     01H
01A0'   CA F01C'                JZ      MONIT
01A3'   C3 0100'                JMP     ISISII
                        ;BUCLA TEST MEMORIE
01A6'   24              LOOP:   INR     H
01A7'   C8                      RZ
01A8'   7E                      MOV     A,M
01A9'   2F                      CMA
01AA'   77                      MOV     M,A
01AB'   BE                      CMP     M
01AC'   2F                      CMA
01AD'   77                      MOV     M,A
01AE'   C0                      RNZ
01AF'   C3 01A6'                JMP     LOOP
                        ;COPIERE <DE> IN <BC> PE <HL>
01B2'   1A              COPY:   LDAX    D
01B3'   02                      STAX    B
01B4'   13                      INX     D
01B5'   03                      INX     B
01B6'   2B                      DCX     H
01B7'   7D                      MOV     A,L
01B8'   3D                      DCR     A
01B9'   C2 01B2'                JNZ     COPY
```



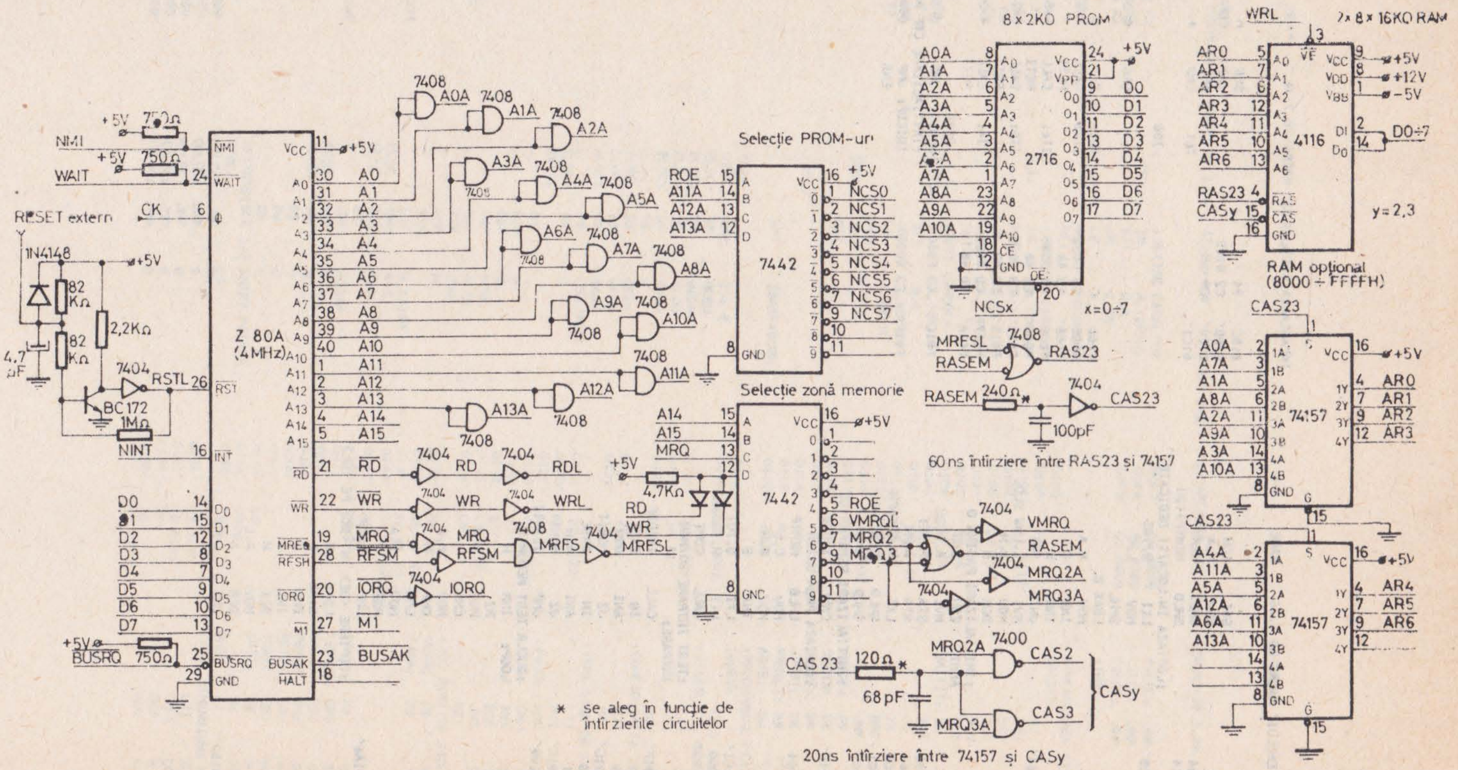
```
01BC    94                      SUB     H
01BD'   C2 01B2'                JNZ     COPY
01C0'   C9                      RET
01C1'                   TOT     EQU     $

                        .Z80

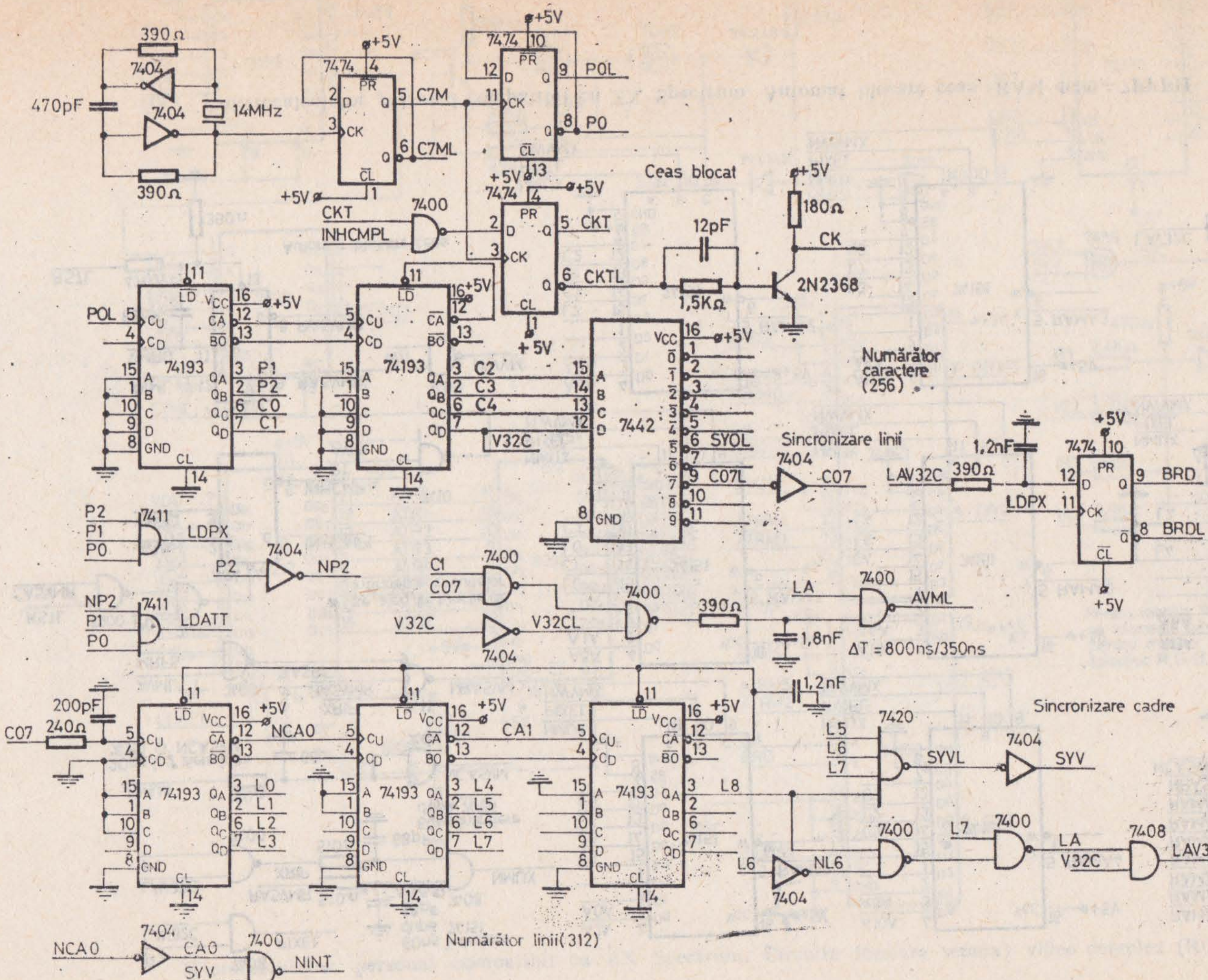
                                ORG     0F000H

F000'                   INI:    DS      8
F008'   CD 0038         IT7:    CALL    7*8
F00B'   ED 4D                   RETI
F00D'   CD 0030         IT6:    CALL    6*8
F010'   ED 4D                   RETI
F012'   CD 0028         IT5:    CALL    5*8
F015'   ED 4D                   RETI
F017'   CD 0020         IT4:    CALL    4*8
F01A'   ED 4D                   RETI
                        .8080
F01C'   C3 F800         MONIT:  JMP     0F800H
                        ;INITIALIZARE CDF ALES
F01F'   C3 0000         INICDF: JMP     00000H
                                END
```

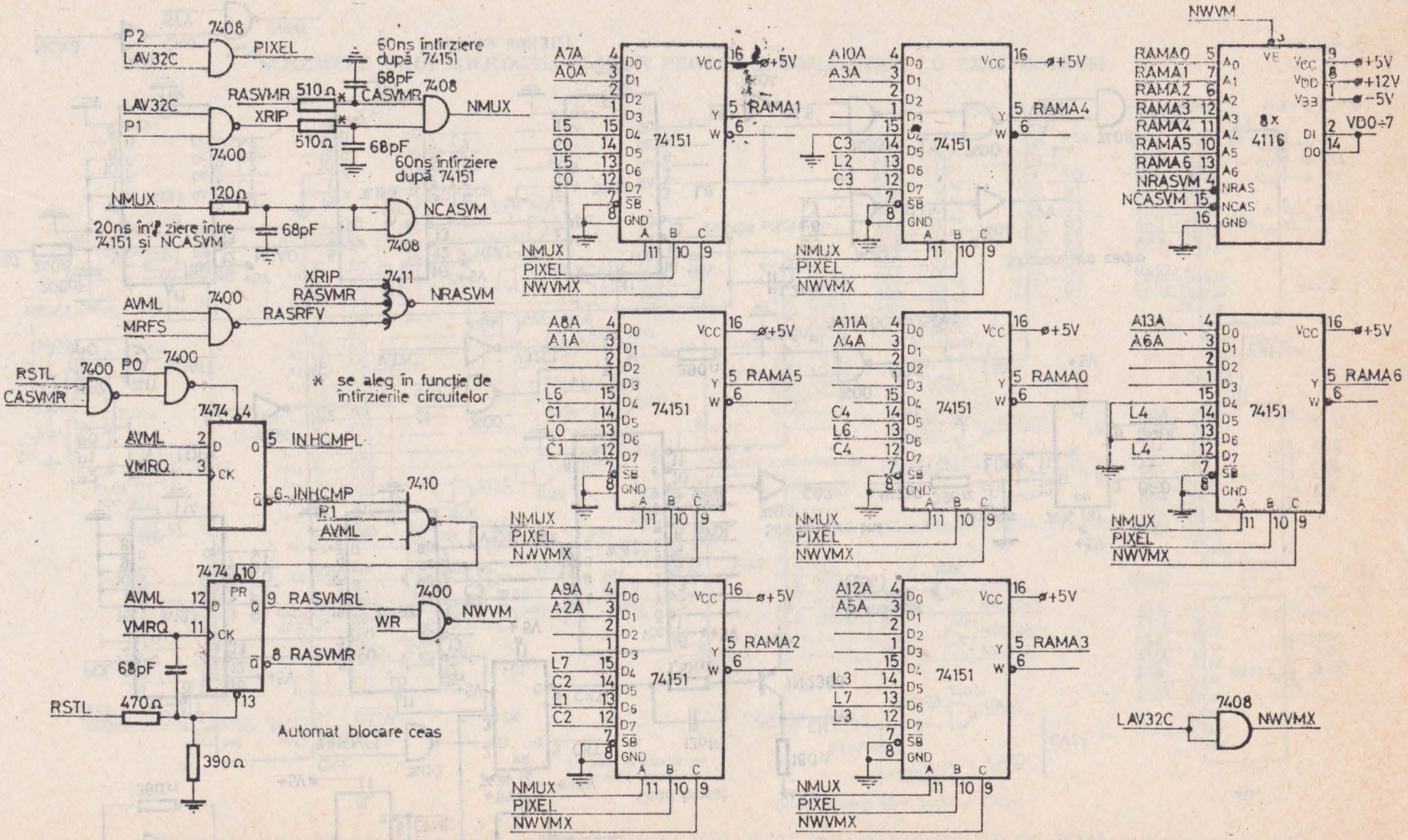
SCHEMELE UNUI MICROCALCULATOR PERSONAL COMPATIBIL CU ZX SPECTRUM



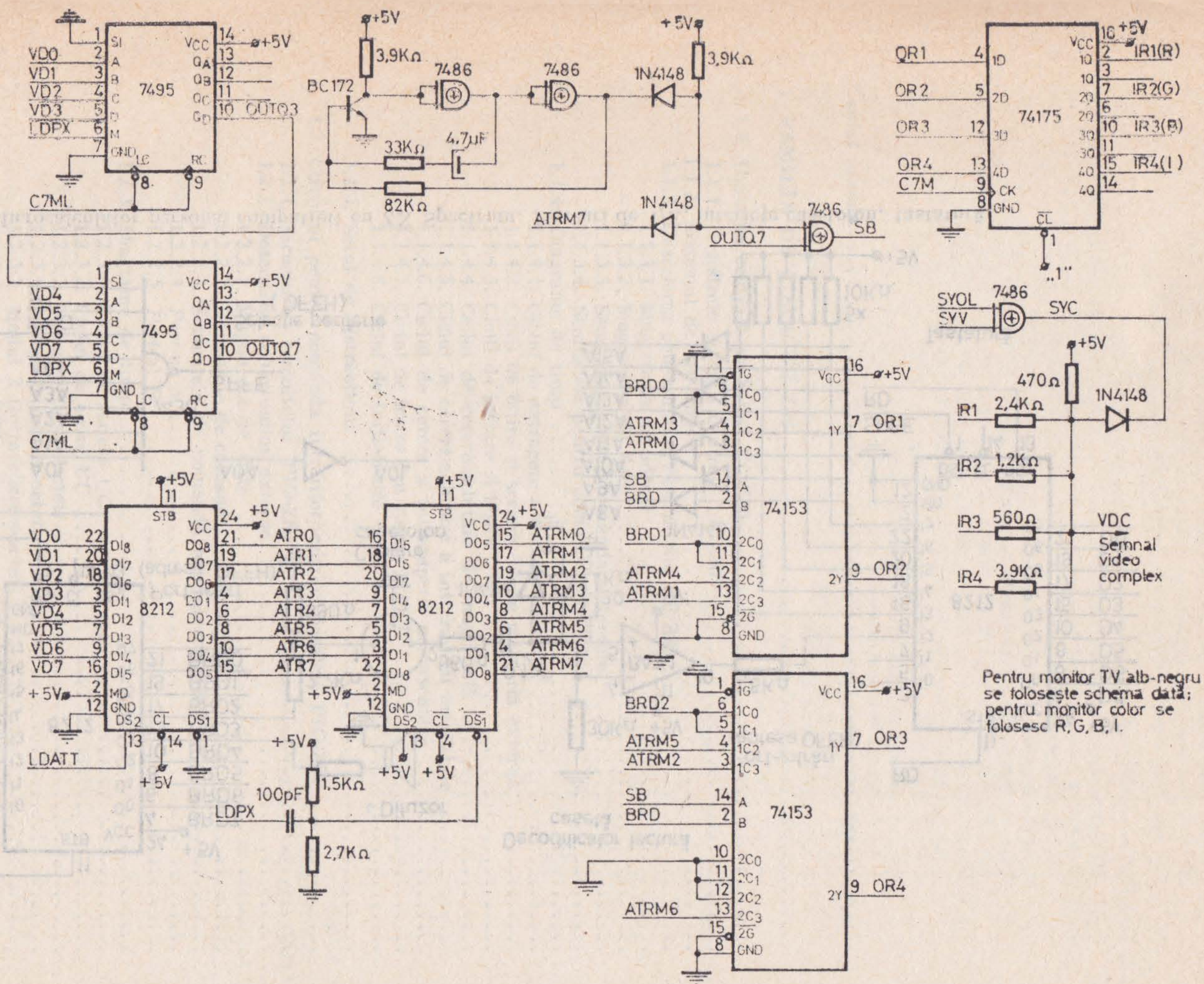
Microcalculator personal compatibil cu ZX Spectrum. Z80, RAM opțional, selecții PROM, RAM, inițializare



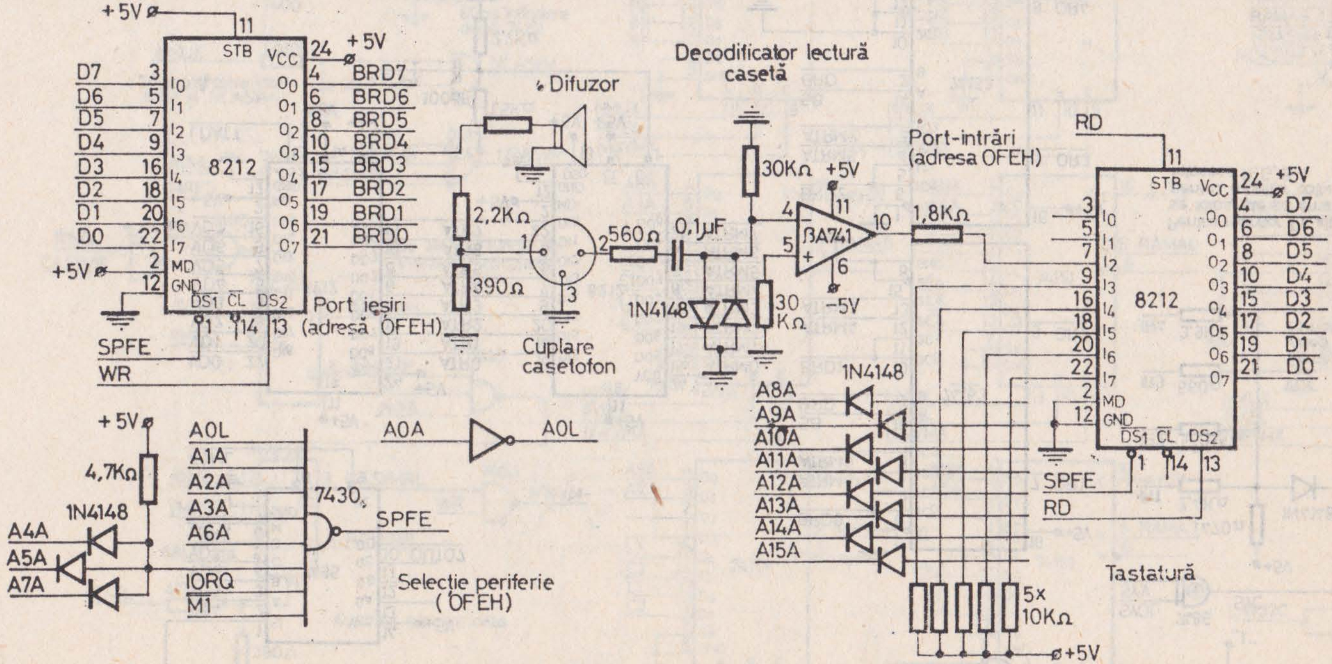
Microcalculator personal compatibil cu ZX Spectrum. Ceas, numărătoare de caractere și linii



Microcalculator personal compatibil cu ZX Spectrum. Automat blocare ceas, RAM 4000÷7FFFH



Microcalculator personal compatibil cu ZX Spectrum. Circuite formare semnal video complex (RGB)



Microcalculator personal compatibil cu ZX Spectrum. *Port*-uri de I/E, interfețe casetofon, tastatură

CUPRINS

Redactor: DUMITRU NICOLESCU
Tehnoredactor: GH. CUCOŞ

Bun de tipar: 12.11.1986
Coli de tipar: 18. B 10186

Tiparul executat sub comanda
nr. 1275 la
Întreprinderea poligrafică
„13 Decembrie 1918".
str. Grigore Alexandrescu nr. 89-97
Bucureşti,
Republica Socialistă România

EDITURA MILITARĂ

Lei 23